These notes were compiled by Octavian Blaga in spring 2017 (thanks Octavian!)

Introduction

Should I buy stock?

- What other stock do you have in your portfolio?
- What do you think will predict stock prices?

How does information affect the movement of the stock price?

- How to build software to analyze and visualize these relationships
- Details about how the stock exchanges work

Machine Learning algos that can be used to build real trading strategies.

Mini-courses:

1. Manipulating Financial Data in Python
    - Read historical financial data into python and manipulate it using powerful statistical algorithms
2. Computational Investing
    - Algorithms, methods and models used by hedge funds and investment banks to manipulate and work with financial data
3. Learning Algorithms for Trading
    - We pull everything together:
        - Take what we learned in the first two mini courses
        - Show how to take that data and use it with machine learning like Q learning and random forests to build trading algorithms

Doc links for Course Software from here http://quantsoftware.gatech.edu/ML4T_Software_Setup:

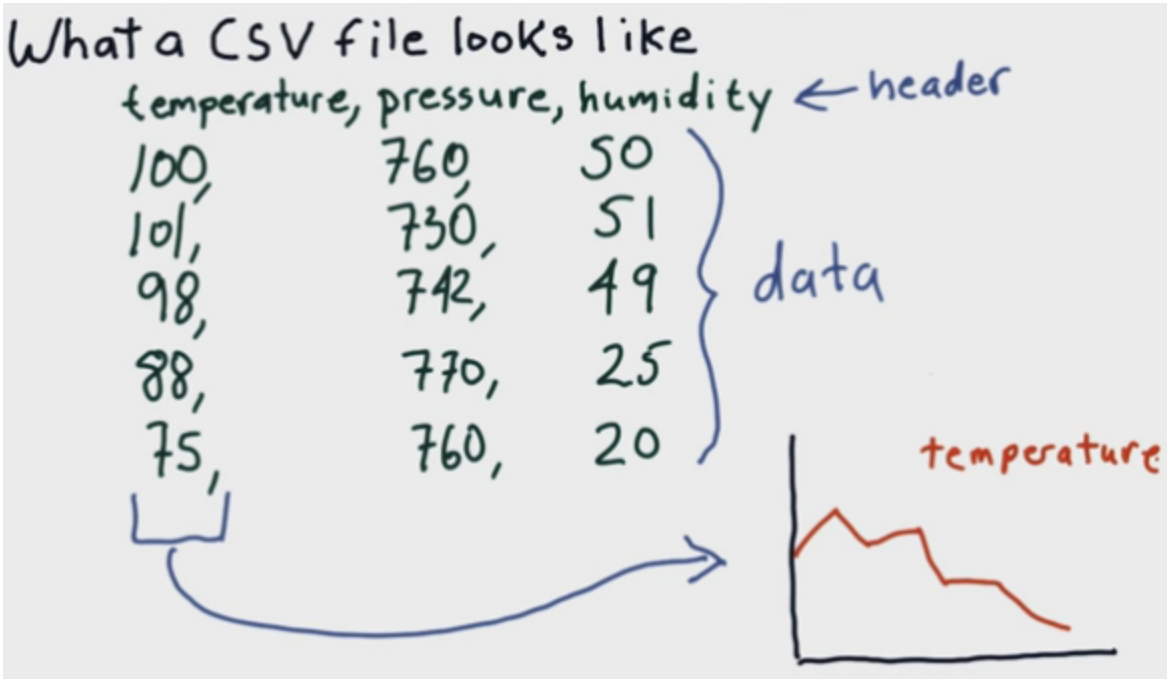
- Python 2.7.6
- NumPy 1.10.2
    - Can only find Docs for 10.1.1
- SciPy 0.16.1
- MatPlotLib 1.5.0
- Pandas 0.17.1
- dateutil 2.4.2
- pytest 2.5.1
    - Can only find Ver 3+, so here's latest

01-01 Reading and Plotting Stock Data

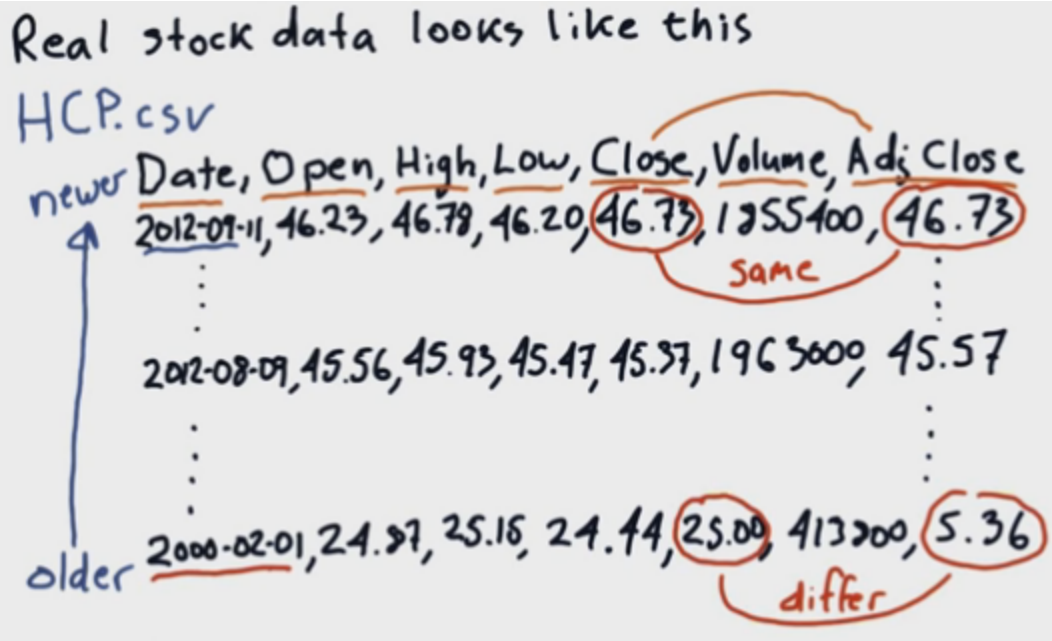
Why Python:

- Strong scientific libraries
- Strongly maintained
- Fast

What a CSV file looks like



Real stock data

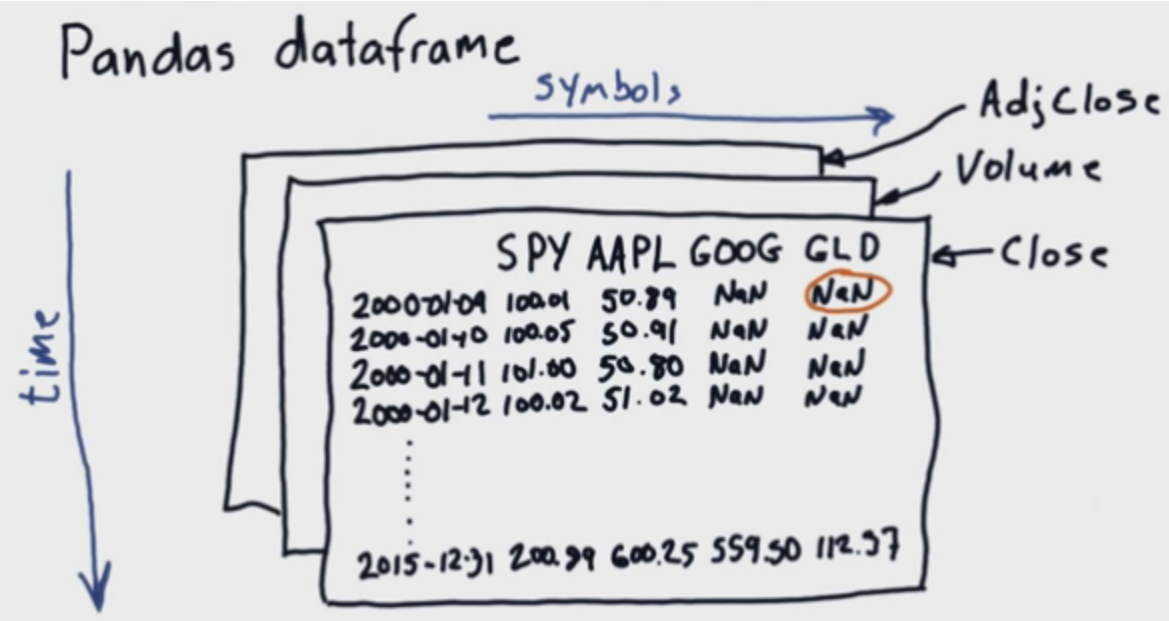


- Close
  - Is the actual price that was reported at the exchange when the stock closed for that day
- Adjusted Close
  - The number that the data provider generates for us and it's adjusted for certain things like stock splits and dividend payments

Pandas

Created by Wes McKinney at a hedge fund called AQR

Data Frame



Example financial CSV

- pd.head() prints the top 5 lines
- pd.tail() the last
- pd.max()

01-02 Working With Multiple Stocks

- **252 days in 2014 when stocks were traded**
    - Must factor in weekends and holidays that occasionally differ from the official US ones

Building a Data Frame (df)

- We use SPY data as a reference between the dates in there are a good reference for when stocks were traded (and it excludes weekends and holidays)

NaNs

- df.dropna() removes rows with NaN values
- df.dropna(subset='COLNAME') removes rows with NaN values in specified column

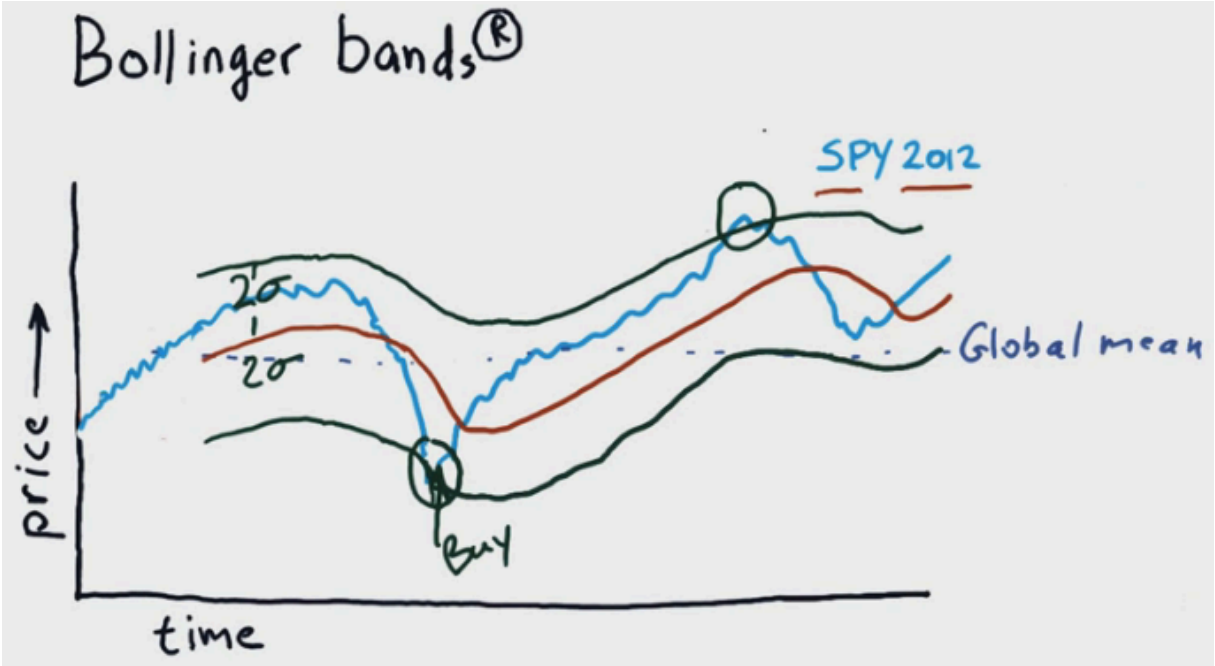
We're doing a join

- Join is much like database/sql joins
- df.join defaults to left join
- df.join(otherdf, how='inner') does inner join, only includes data where both exist in df and other df
- It eliminates all the bad dates

01-04 Statistical Analysis of Time Series

Bollinger Bands (trademark)

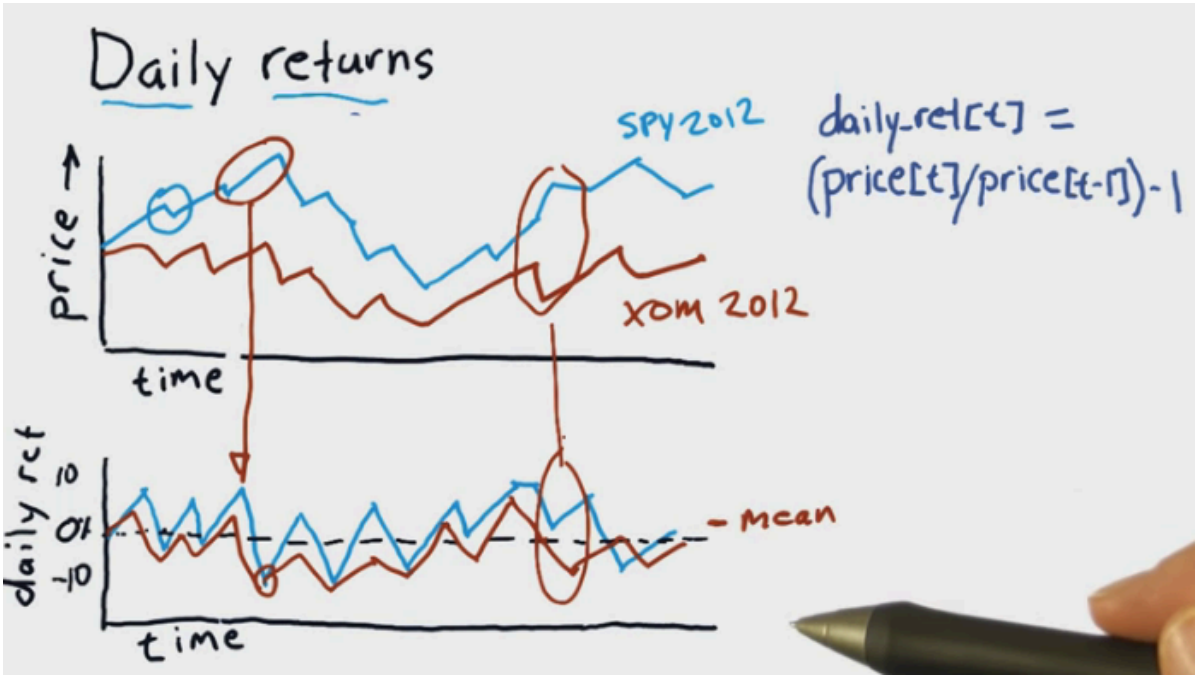
- You add 2*sigma (standard deviations) above and subtract below



Time to buy when price recovers from below -2sigma and time to sell when price falls from above 2sigma.

Daily Returns

How much did the price go up and down on a particular day?
(Today's stock price / Yesterday's stock price) - 1



- **When SPY went up, XOM went down**

df[:-1].values accesses the underlying numpy array. It is needed because pandas by default will divide same indices, whereas Numpy is first from start in one is divided by first from start in second.

daily_ret[1:] = (df[1:]/df[:-1].values) - 1

daily_ret = (df / df.shift(1)) - 1

daily_ret.ix[0, :] = 0 #set daily returns for row 0 to 0 because panda leaves 0th row full of nans

Np.empty uses existing memory which is not guaranteed to be initialized to 0

Cumulative Returns

(Today's price / Price at the beginning) - 1

The -1 is is to eliminate that initial division result which is with respect to the first value such that the plots center around 0.

01-05 Incomplete Data

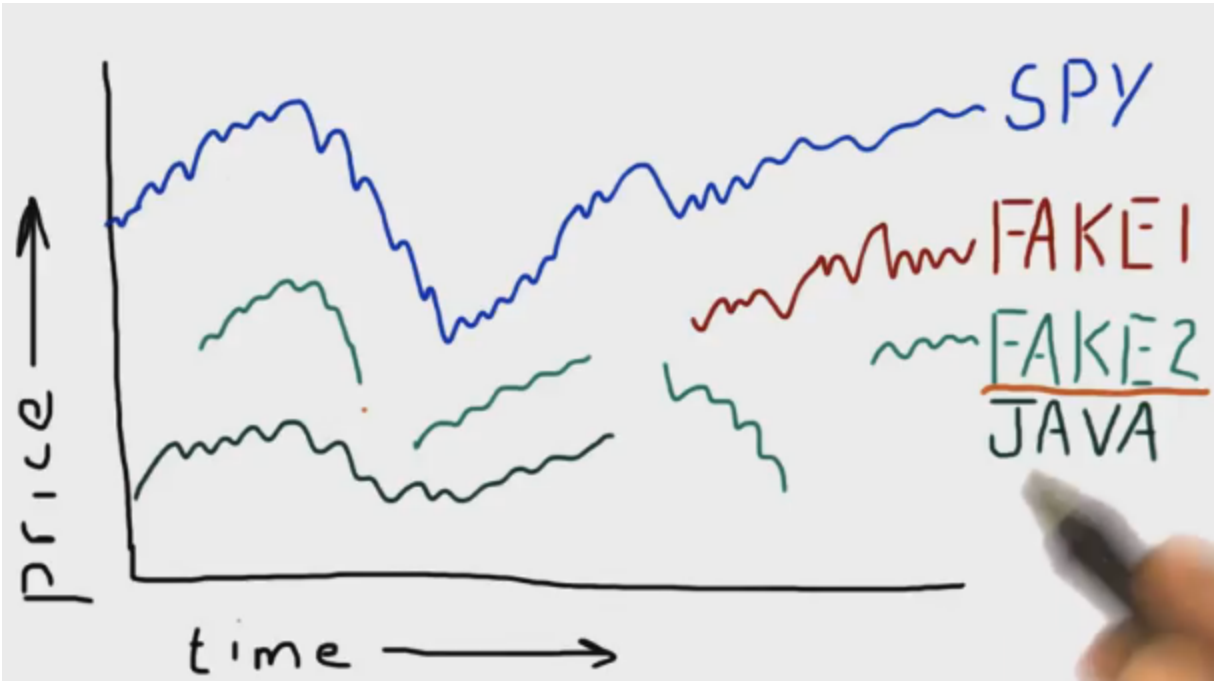
Pristine Data? No.

- What people think
  - Perfect data recorded minute by minute
  - No gaps or missing data points
- Reality
  - Data is an amalgamation from many sources
    - A stock may trade at different prices in NY Stock Exchange, another price as NASDAQ
      - There is no single price for any stock
  - Not all stock trade every day
    - Stocks come into existence and there's values for them and before there was no data
    - Sometimes stocks go out of existence and suddenly they quit existing and there's no data for them going forward
  - Sometimes stocks will be trading, data will be missing, then it starts trading again

An ETF or Exchange-Traded Fund is a basket of equities allocated in such a way that the overall portfolio tracks the performance of a stock exchange index. ETFs can be bought and sold on the market like shares.

For example, SPY tracks the S&P 500 index (Standard & Poor's selection of 500 large publicly-traded companies).

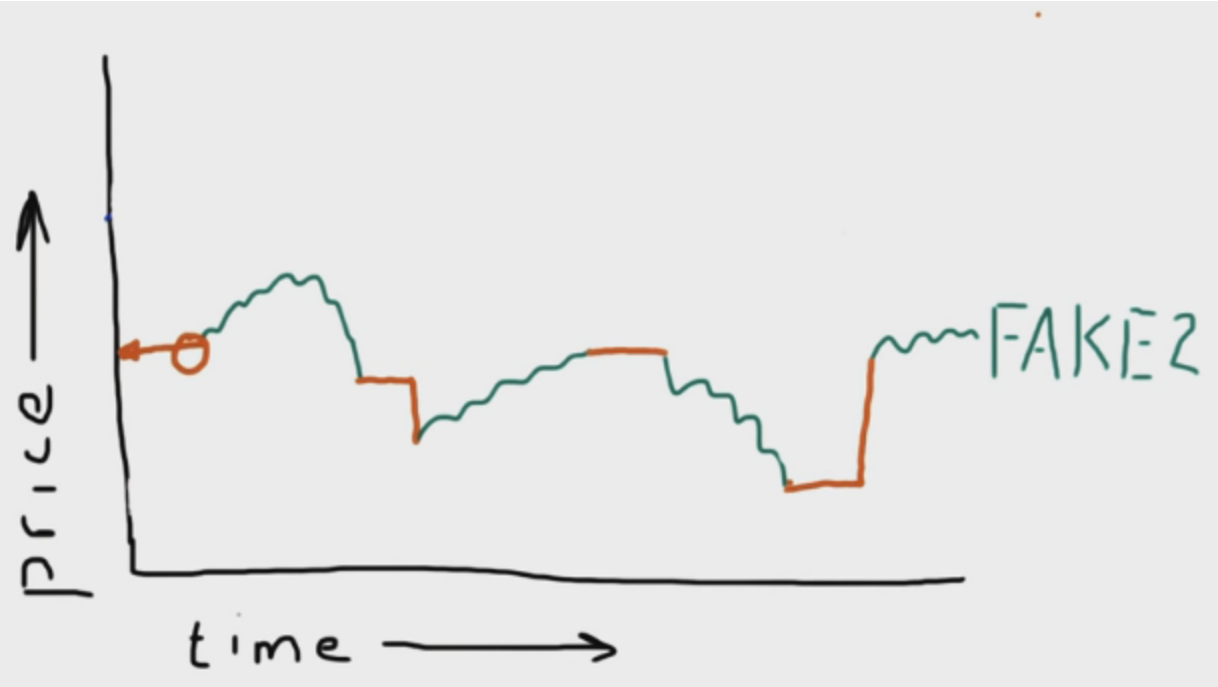
You will learn more about ETFs and other funds in the second mini-course.



How to deal with Incomplete Data?

You never interpolate!

First Fill Forward (takes care of any graphs) and then second Fill Backward (takes care of any time which is less than the first known data point)



Pandas has a function called df.**filna**

df_data.fillna(method='ffill', inplace=True)
df_data.fillna(method='bfill', inplace=True)

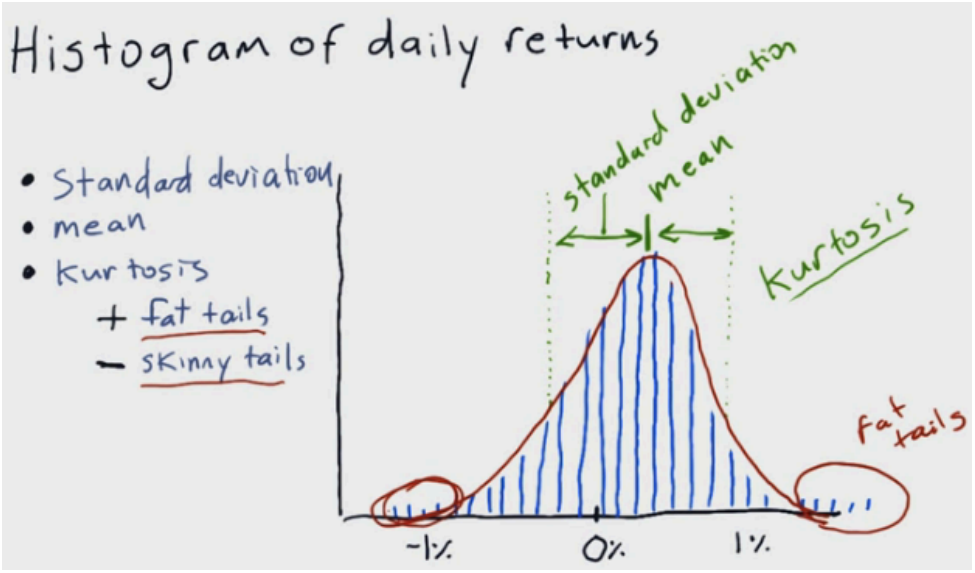
01-06 Histograms and Scatter Plots

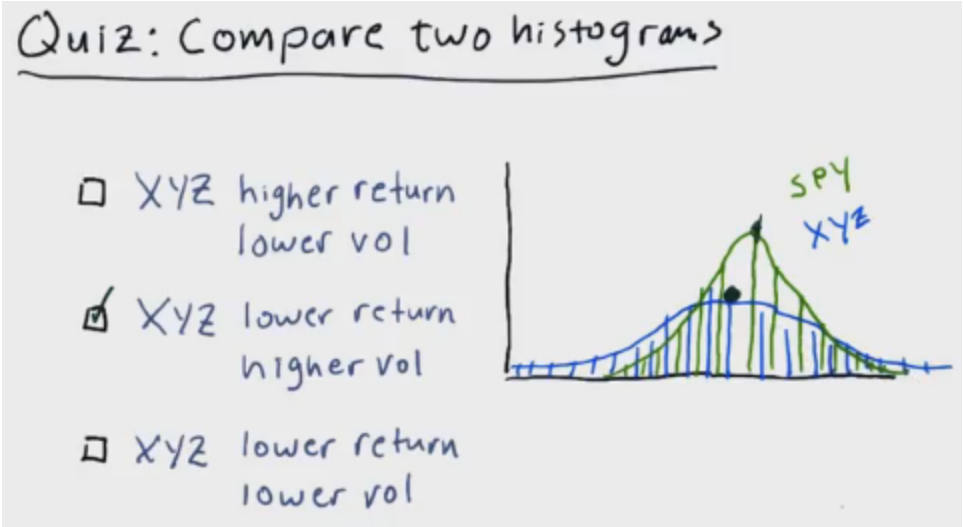
Histogram of Daily Returns

Kurtosis

It means curved or arching

- Tells us about **the tails of a distribution**
- If we assume that our histogram is similar to a gaussian Distribution or normal distribution, then the measure of Kurtosis tells us how different our histogram is from that of the traditional distribution.
    - We have fat tails and what this means is that there are occasional and more frequent large excursions than would happen if we had a regular Gaussian distribution
- If we measure a positive Kurtosis (fat tails) it means that there are more occurrences out in these tails than there are in the normal distribution
- If we measure a negative Kurtosis (skinny tails) it means that there are fewer occurrences out in these tails than there are in the normal distribution



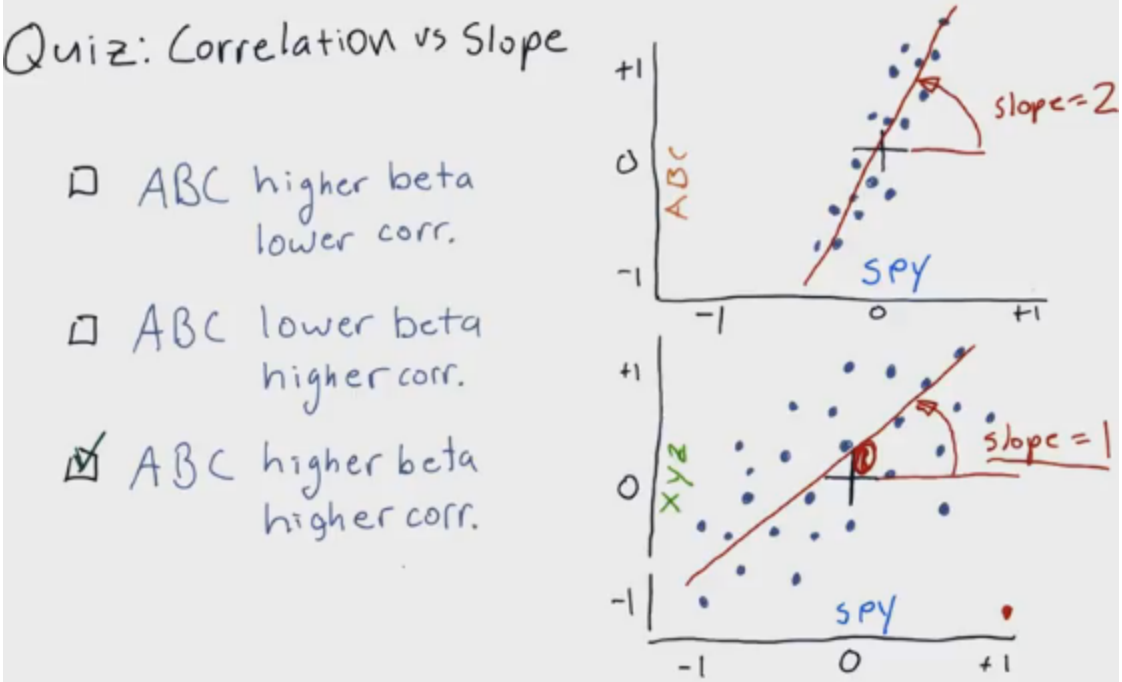


Scatter Plots (the daily returns of that stock with respect to SPY)

Beta - how reactive is the stock to the market. If the slope (Beta) is 1 it means that if the market goes up 1%, that stock also goes up 1%. If beta is 2, if the market goes up 1%, the stock goes up 2%.
Alpha - That stock on average is performing a little bit better than the S&P500 (SPY) every day if **Alpha** is positive. If it's negative it means it's returned a little less than the market overall.

The slope (Beta) is not Correlation!

Correlation is how tightly these individual points fit that line. So you could have a shallow slope with a higher correlation, or you can have a steeper slope and the data fitting the line at a higher correlation.



As we have seen in this lesson, the distribution of daily returns for stocks and the market look very similar to a Gaussian, this property persists when we look at weekly, monthly and annual returns as well. If they were really Gaussian we'd say the returns were normally distributed.
In many cases in Financial Research we assume the returns are normally distributed but this can be dangerous because it ignores **kurtosis or the probability in the tails**. In the early 2000s, investment banks

built bonds based on mortgages. They assumed that the distribution of returns for these mortgages was normally distributed and on that basis they were able to show that these bonds had a very low probability of default but they made two mistakes - FIRST they assumed that the return of each of these mortgages was independent and TWO that this return would be normally distributed. Both the assumptions proved to be wrong as massive numbers of homeowners defaulted on their mortgages. It was these defaults that precipitated the great recession of 2008.

01-07 Sharpe Ratio and Other Portfolio Statistics

- This lesson is about evaluating the performance of portfolios and accordingly, portfolio managers.
- **We'll define a portfolio as an allocation of funds to a set of stocks.**
- **Sharpe Ratio is RISK ADJUSTED RETURN**
    - **Larger Sharpe ratios are better**

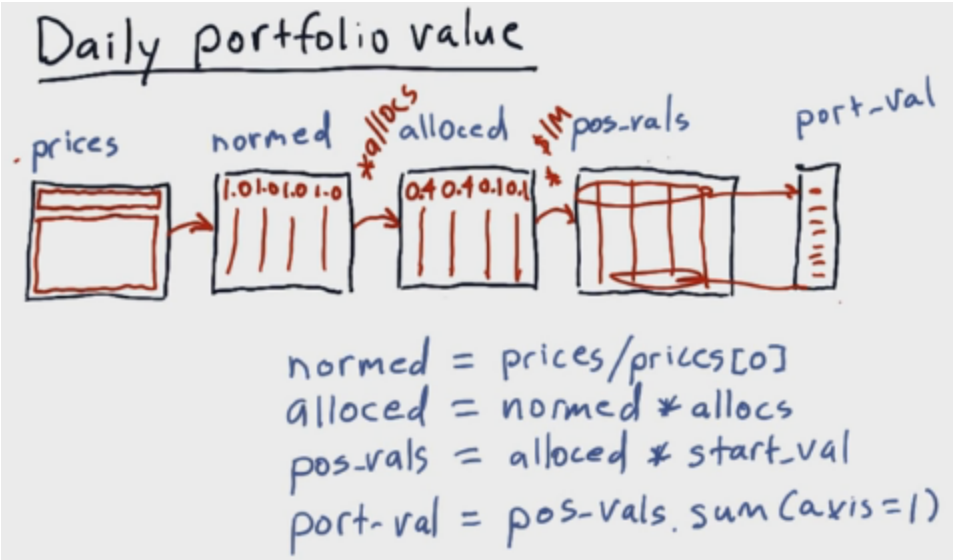
Buy-and-Hold Strategy

Where we invest in a set of stocks with a certain allocation and then observe how things go moving forward. We'll assume the allocations sum up to 1.0

Daily Portfolio Value

```
Start_val = 1000000
Start_date = 2009-1-1
End_date = 2011-12-31
Symbols = ['SPY', 'XOM', 'GOOG', 'GLD']
Allocs = [0.4, 0.4, 0.1, 0.1]
```

How do we calculate the total value of the portfolio day by day?



1. `normed = prices/prices[0] #normalized prices`
2. `alloced = normed * allocs`
3. `pos_vals = alloced * start_val #the amount of cash allocated to each asset, position values`
4. `port_val = pos_val.sum(axis=1) #daily total value of a portfolio`

Portfolio Statistics

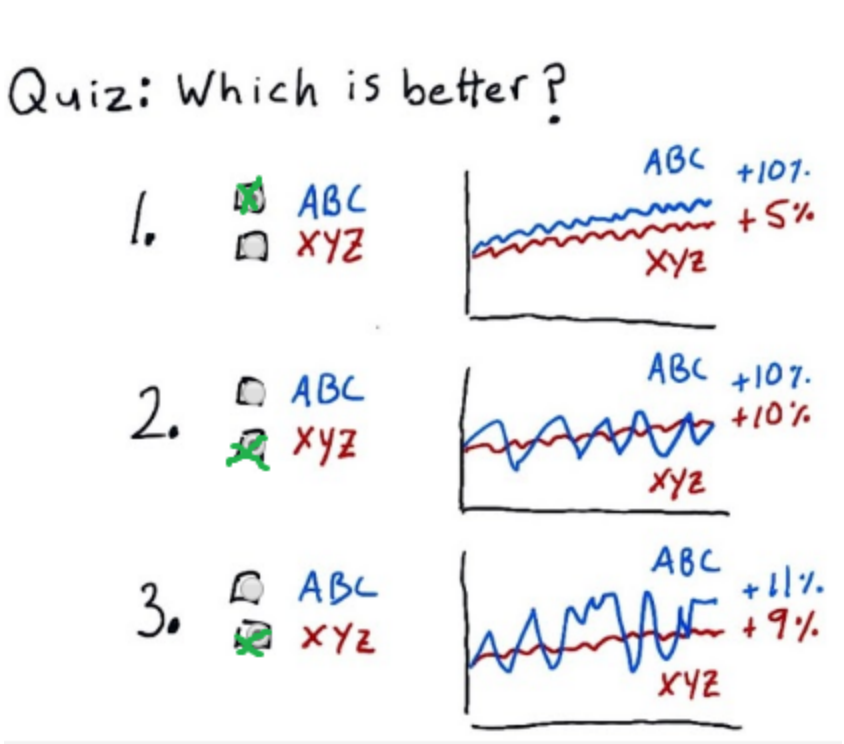
The daily return value on the first day is always 0 because there is no change on that day and we want to exclude that value from any calculations we do across all daily returns.:

- **daily_rets = daily_rets[1:]**
- **cum_ret = (port_val[-1] / port_val[0]) - 1**
    - **Cumulative return is just a measure of how much the value of the portfolio has gone up from the beginning to the end.**
        - So to calculate that we take the last val, which is port-val of -1
- **avg_daily_ret = daily_rets.mean()**
- **std_daily_ret = daily_rets.std()**
- **sharpe_ratio**

Sharpe Ratio

William Sharpe

- The idea for the Sharpe ratio is to consider our return or our rewards in the context of risk
    - Most finance folks consider risk to be standard deviation or volatility



Correct!

Good job!

1. Both stocks have similar volatility, so ABC is better due greater returns.
2. Here both stocks have similar returns, but XYZ has lower volatility (risk).
3. In this case, we actually do not have a clear picture of which stock is better!

-
- You can't just look at 1. And listen to what your gut says, you need a way to qualitatively measure this betterness and this is what the Sharpe ratio is all about

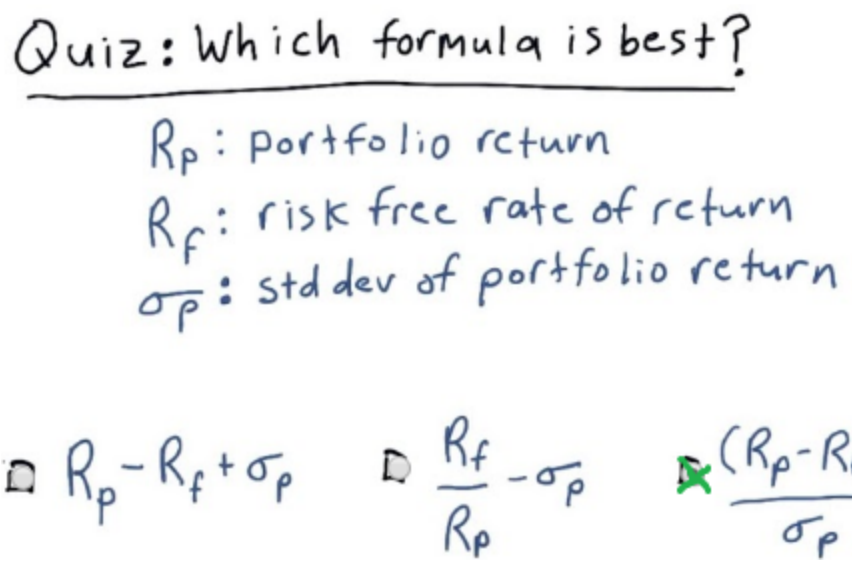
Sharpe ratio is a metric that adjust return for risk

- risk adjusted-return
- all things being equal
    - Lower risk is better
    - Higher return is better

- Sharpe ratio (SR) also considers
    - Risk free rate of return (which lately is about 0% since 2000s)

Which formula is best?

- **Rp: portfolio return**
- **Rf: risk free rate of return**
- σ_p **: std dev of portfolio return**



Correct!

That's right!

The value of a portfolio is directly proportional to the return it generates over some baseline (here risk-free rate), and inversely proportional to its volatility.

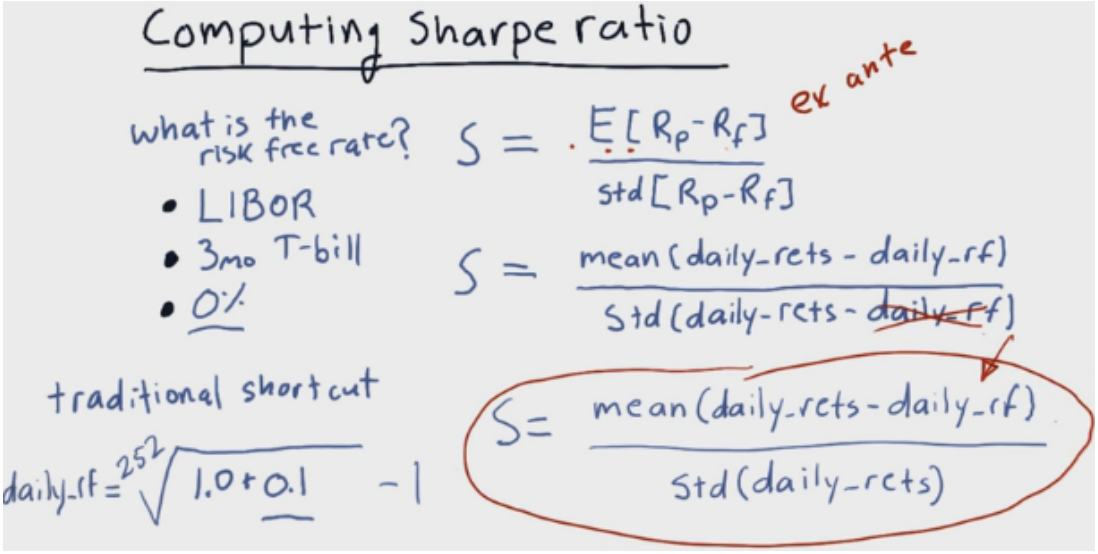
Also notice that this form subtracts like quantities (returns), unlike the other two.

- **Keep an eye out for forms that subtract unlike quantities!**
    - **As things become more volatile (sigma_p goes up), the ratio goes down**
    - **As the risk free metric increases, the value of our metric, the Sharpe ratio decreases**

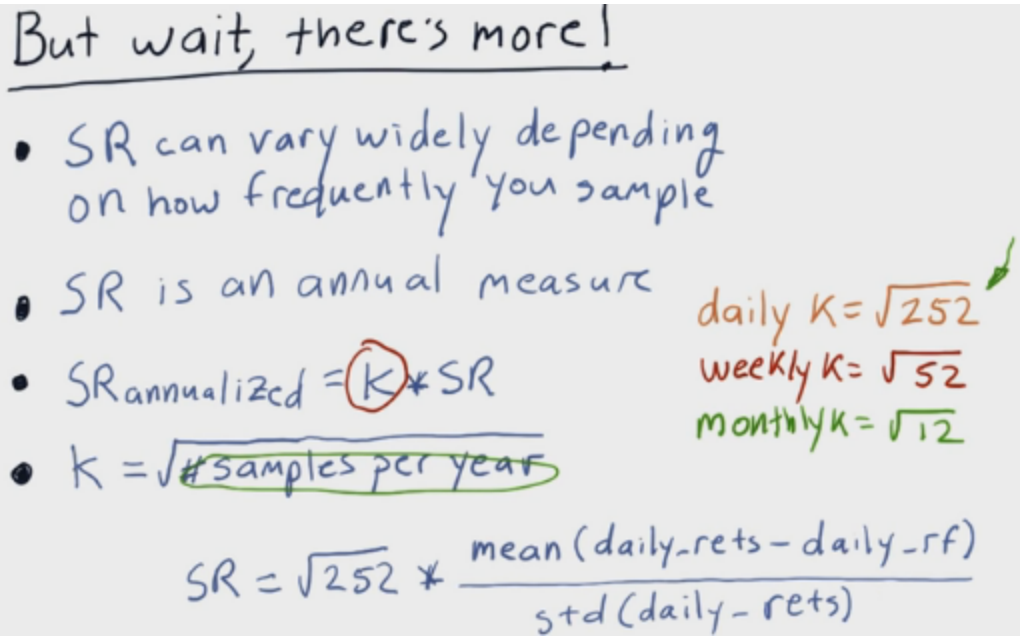
$$S = \frac{E[R_p - R_f]}{std[R_p - R_f]}$$

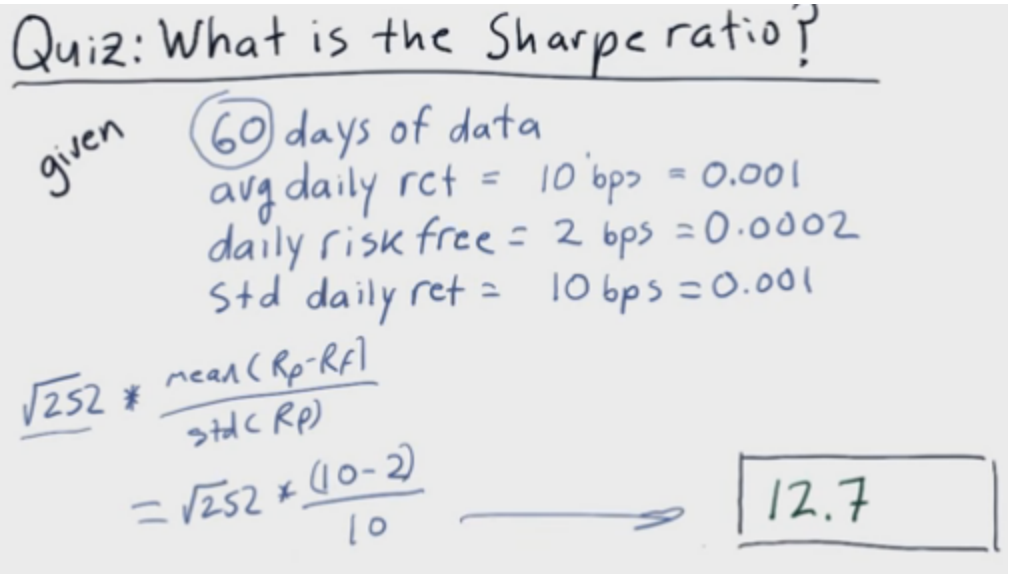
- It's the expected value of the return on a portfolio, minus the risk free rate of return, divided by the standard deviation of that same difference
- This is the ex ante formulation, meaning because we're using **Expected** is a forward looking measure of what the Sharpe ratio should be
- To calculate this in reality, we need to look back at those values, so for instance the expected value of this difference is just simply the mean of what that difference was over time
- In python using historical data: $S = \frac{mean(daily\_rets - daily\_rf)}{std(daily\_rets - daily\_rf)}$
    - **What is this Risk free rate?**
        - **LIBOR:** London Interbank Offer Rate
        - **3mo T-bill:** Three Month Treasury Bill
        - **0%**, and finally, a value that people have been using a lot over the last few years is 0%
            - 0 % is a good approximation of the risk free rate, *traditional shortcut*

- **Usually it's not given on a daily basis, and it's a percentage on the annual or six months basis so you can convert that annual amount into a daily amount using this simple trick**
  - Let's suppose our risk free rate is 10% per year or 0.1. That means if we start at the beginning of the year with a value of 1.0 at the end of the year we'll have a value of 1.1. What is the interest per day that let's us get to that value:



- SR can vary widely depending on how frequently you sample, yearly you get one number, monthly a different number and if you sample daily you'll get still another number.
- The original vision for the Sharpe ratio is that it's an **annual measure**. So if we're sampling at frequencies other than annually we need to add an adjustment factor to make it all work out
  - SRannualized = k*SR, k is the adjustment factor
    - $k = \sqrt{\#samples\ per\ year}$
      - For daily data there are 252 trading days per year so k is $\sqrt{252}$
        - Even if we traded for 89 days, we use 252 because we're sampling daily
          - It's the frequency at which we sample that affects the value of k
      - For weekly data there are 52 trading weeks per year so k is $\sqrt{52}$
      - For monthly data there are 12 trading months per year so k is $\sqrt{12}$





- bps, basis points are tenths of a percentage, 10 basis points, bps are 1%

01-08 Optimizers

What is an optimizer? An optimizer is an algorithm that can do the following:

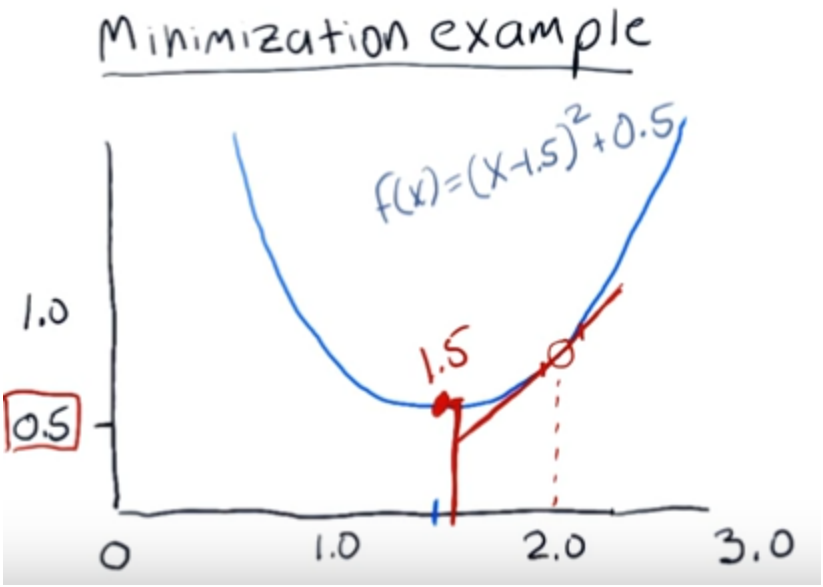
- **It can find minimum values of functions**
    - **f(x)=x^2+x^3+5**
        - For what value of x is the above minimized?
- **Build parameterized models based on data**
    - Find the parameters for parameterized from data
- **Refine allocations to stocks in portfolio**
    - You can decide what percentage of funds should be allocated to each stock using the optimizer

How to use an Optimizer

1. **Provide a function to the minimizer - define a function that you need to minimize**
    - **f(x) = x^2+.5**
2. **Provide an initial guess for x that might be close, or a random value**
3. **Call the function over and over (until it converges)**

Example

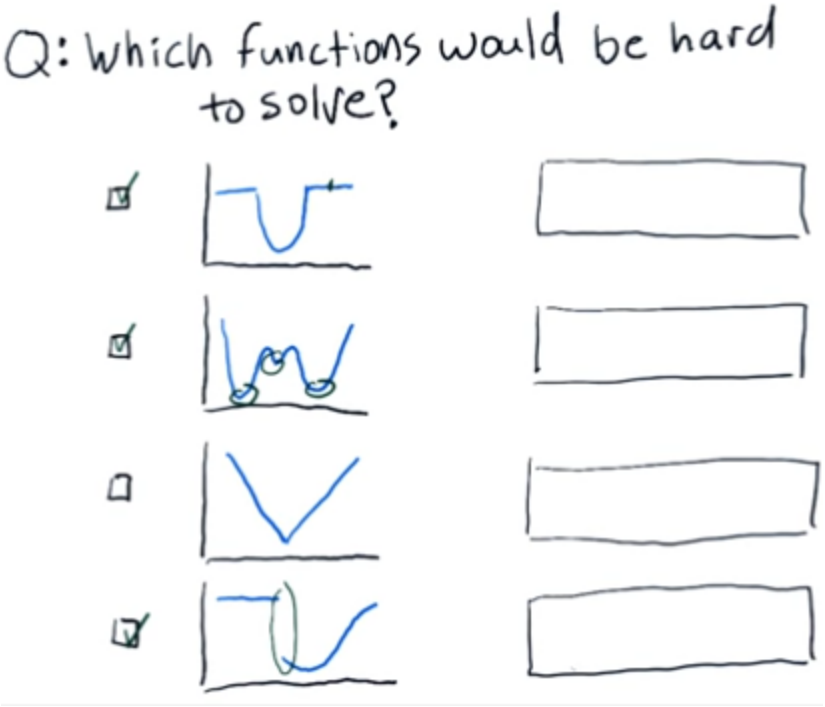
- Start with a guess of 2.0 and figure out the minimum
- It marches downhill, gradient descent is a minimizer

- 


```python
"""Minimize a function"""

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import scipy.optimize as spo

def f(X):
    """Given a scalar X, return a real value: f(x)"""
    Y=(X-1.5)**2+0.5

    print "X={}, Y={}".format(X,Y)
    return Y

def test_run():
    Xguess=2.
    min_result=spo.minimize(f, Xguess, method='SLSQP', options={'disp':True})
    print "Minima found at:", "X={}, Y={}".format(min_result.x, min_result.fun)

    #Plot
    Xplot=np.linspace(0.5, 2.5, 21)
    Yplot=f(Xplot)
    plt.plot(Xplot, Yplot)
    plt.plot(min_result.x, min_result.fun, 'ro')
    plt.title("Minima of the objective function")
    plt.show()

if __name__=="__main__":
    test_run()
```

Which functions would be hard to solve?

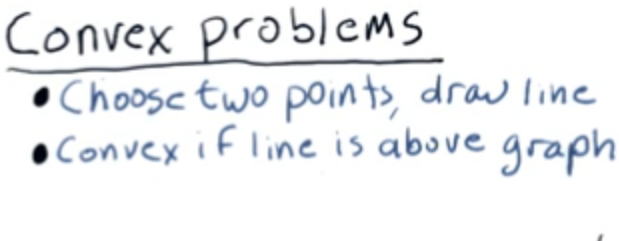
- Functions with flat areas are hard to compute the solve on,
- Functions with multiple minima
- Functions with discontinuities



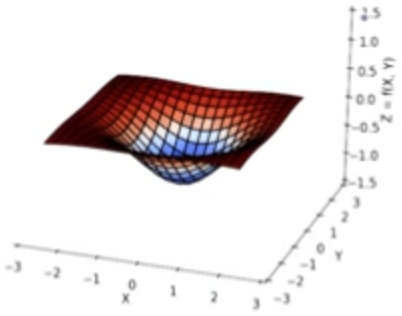
- **The minimizers are guaranteed to find a minima, not THE mimina**

Convex Problems

- A particular class of problems that are the easiest for optimizers to solve
- **A real valued function f(x) defined on an interval is called convex if the line segment between any two points on the graph of the function lies above the graph**
  - Choose any two points and draw a line
    - **Convex if line is above graph**

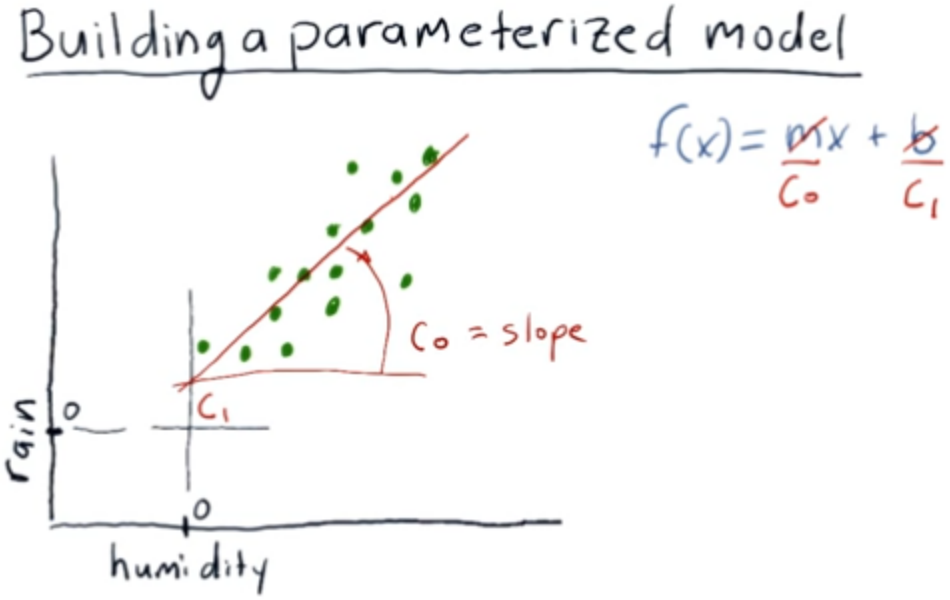


  - In order for the function to be convex, it must have only one local minima which is also the global minima
  - We also can't have any flat regions
  - The optimizers can just as easy work in multiple dimensions with gradient descent

- 

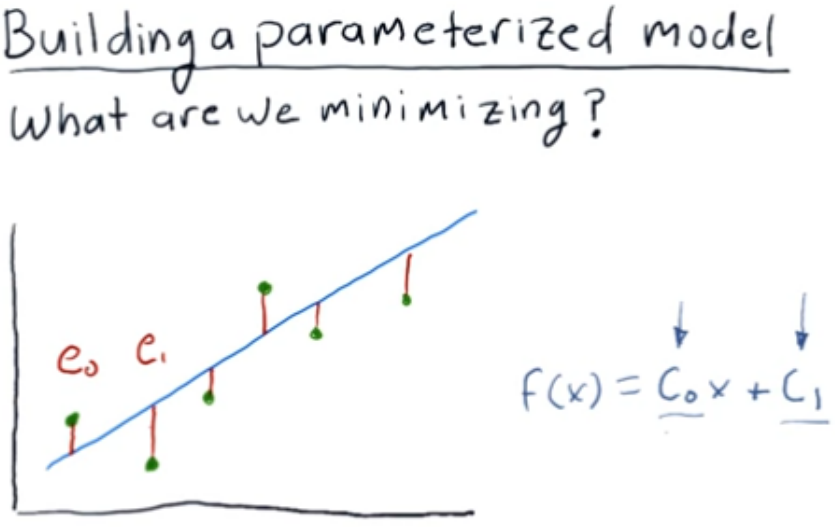
Building a parameterized model (from data)

- **f(x) = mx + b**
  - **m and b are parameters of that model**
  - **Equivalent to f(x) = c0*m + c1**



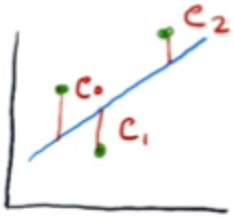
Restating the problem

What are we minimizing?



Q: Which of these metrics would be good?

- ☐ $\sum_i e_i$
- ☑ $\sum_i abs(e_i)$
- ☑ $\sum_i e_i^2$



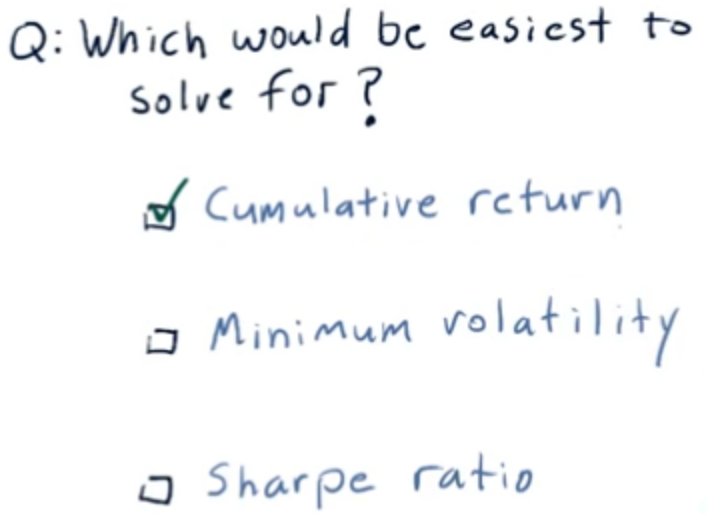
- The sum of ei is not a good one because some of the values are negative and you could end up with a negative error
- ei^2 is famous the squared error

Minimizer finds coefficients

-

01-09 Optimizers : How to optimize a portfolio

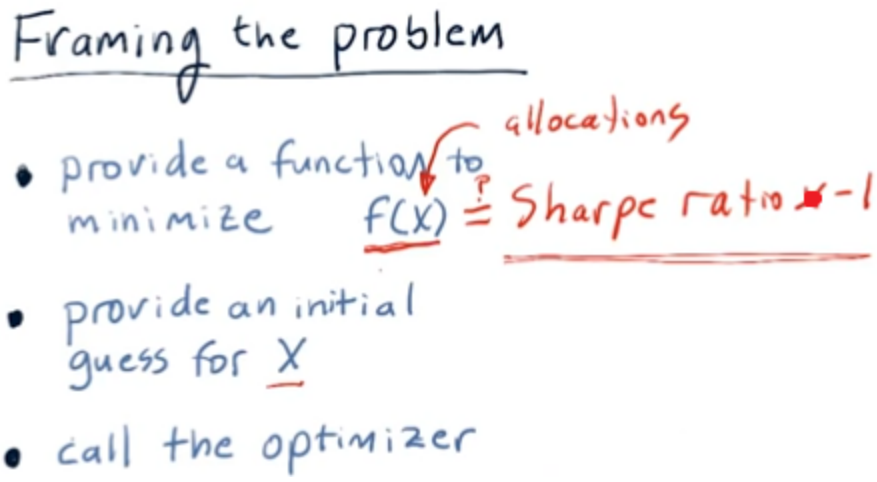
Which would be the easiest to solve for?



- The simplest is Cumulative return because all you have to do is find the single stock that maximized return over that time period
    - The allocation would be 100% to the highest returning stock
- If you're going to optimize for minimum volatility or Sharpe ratio you have to evaluate various combinations of those stocks

Framing the problem

- In order to use an optimizer that minimizes we need to do the following:
    - Provide a function to minimize f(x)
    - Provide an initial guess for x
    - Call the optimizer
- In this case X are the allocations we're looking for in fractional units adding up to a max of 1
    - And we want the optimizer to try different allocations to discover the best set of allocations that optimizes this function
        - We want to optimize for Sharpe ratio, but f(x) is not the actual Sharpe ratio value, we want to optimize for -1 * the largest Sharpe ratio because the available optimizer function is ***minimize***
            - This will find the best allocation or the best value for x to solve this problem
                - x can have multiple dimensions and each of the dimensions is the percentage of funds to allocate to each of those stocks



Ranges and Constraints

- Ranges: Limits on values for x
  - **0.0-1.0**
- Constraints: Properties of the values of x that must be true:
  - $\sum_{i=0}^{3} abs(x_i) = 1.0 = 100\%$
  - In other words the total allocation must add to 100% (1.0)

02-01 So you want to be a hedge fund manager?

- **First lesson of computational investing**
    - Your motivation is probably compensation which depends on:
        - What kind of fund you manage
        - How well does the fund perform

Types of Funds

- **ETF - exchange-traded funds**
    - Are very much like stocks
    - Buy/sell like stocks
    - Baskets of stocks
        - Sometimes they represent other **instruments** like bonds and so on
        - It's very well known and they publish what it is they're holding
    - Transparent
        - Very transparent
        - Very liquid
- **Mutual Funds**
    - Buy/sell at the end of day
    - Quarterly disclosure
        - Less transparent
            - Since the last disclosure you don't know what they may have bought or sold
            - But still somewhat transparent because they have stated goals and what they're trying to achieve
- **Hedge funds**
    - Buy/sell by agreement
    - No disclosure
    - Not transparent
        - Usually describe to clients what their strategy is

- LIQUID
    - **The ease with which one can buy or sell shares in a particular holding**
- LARGE CAP
    - **CAP means capitalization or in other words how much is the company worth according to the number of shares that are outstanding times the price of the stock**
        - **#shares * price**

What type of fund is it?

- If it has symbols is not a hedge fund.
- **Mutual funds have 5 letters**
- **ETFs have 4 or 3**
- Hedge funds may have 100 investors while the other one have thousands if not millions of investors.

QUIZ What type are the below?

- **VTINX**
  - Vanguard Target Retirement Income Fund Investor Shares
  - Mutual Fund
- **DSUM**
  - Chinese Yuan Dim Sum Bond Portfolio
  - ETF
- **FAGIX**
  - Fidelity® Capital & Income Fund
  - Mutual Fund
- **Bridgewater PureAlpha**
  - **Doesn't have symbols and it's a hedge fund**
  - Hedge Fund
- **SPLV**
  - SPLV - PowerShares S&P 500 Low Volatility Portfolio
  - Mutual Fund

Incentives: How are the managers compensated?

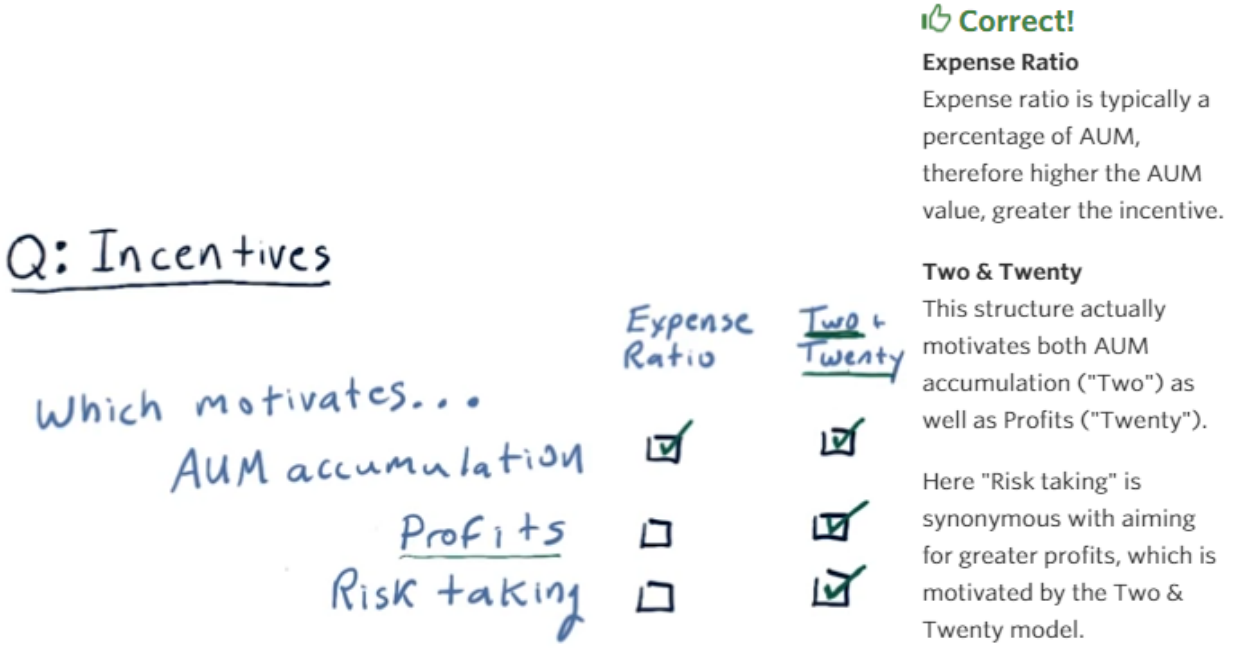
Assets Under Management: **AUM**

- **How much money is being managed by the fund?**
- **There's a % of the AUM that's part of the compensation**

- ETFs
  - Are compensated according to an **Expense Ratio that ranges from 0.01% (one bip) to 1.00%**
  - ETFs are tied to an index like the S&P500, they require less skill manage than Mutual Funds
- Mutual Funds
  - **Expense Ratio ranging from 0.5% to 3.00%**
  - They require more skill to manage than ETFs
- Hedge Funds
  - **"Two and Twenty" - 2% of AUM + 20% of the profits**

Two and Twenty

- $100,000,000
- 15% return (because of your skill)
  - So the compensation is:
    - Two: $100M * 0.02 = $2M
    - Twenty: $15M * 0.2 = $3M
    - Total = $5M



- **AUM accumulation**
  - Most strongly motivates that but 2-20 also works because of 2%
  - If you were strictly driven by the expense ratio you're going to be applying most of your energy towards accumulating assets so that you'll have a higher return so as a portfolio manager you'll get more money
- **Profits**
  - ETF and mutual fund managers are not compensated for making profit so Expense Ratio is out of the question
  - 2-20 - they're always going to get the 2% but the cheeze is 20%
- **Risk Taking**
  - 2-20 - they're always going to get the 2% if they don't make a profit but if they take a risk and make a profit then they'll be compensated more

How funds attract investors

Who?

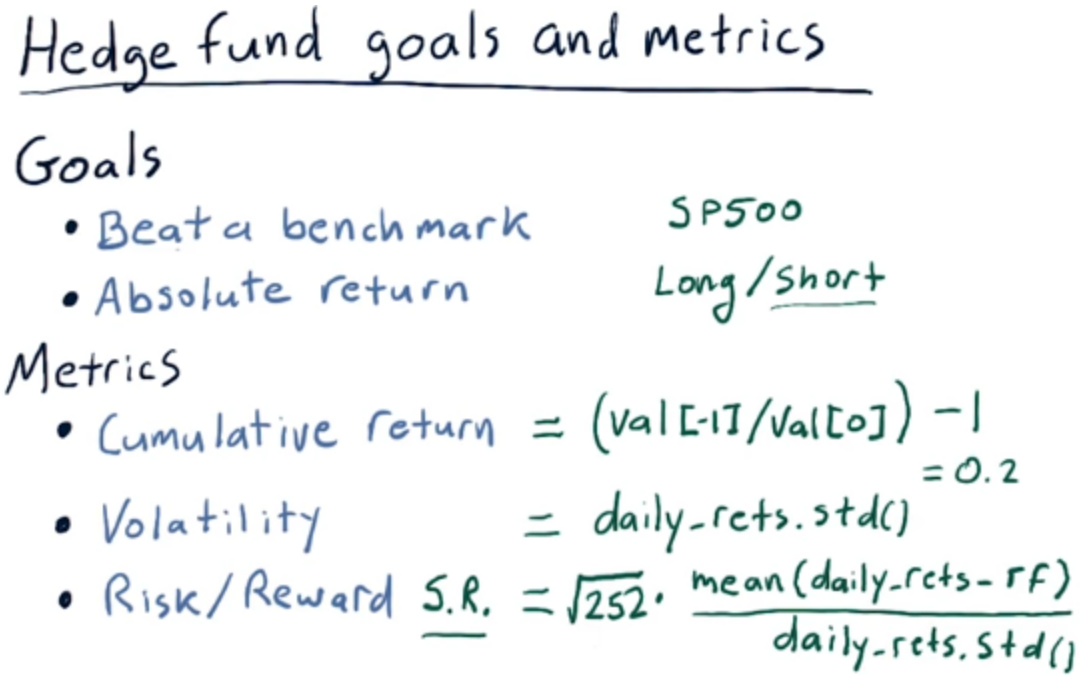
- Individuals
- Institutions
- Funds of funds

Why?

- Track record
- Simulation + **Story**
  - Backtest your strategy
  - Very compelling Story when you don't have a 5 year track record, what do you do?
    - Must have a reason for why this method works and it needs to make sense
    - How your strategy fits within their portfolio:
      - If your strategy is for large cap SP500 stocks and they've already got that covered with another fund, they might not consider you
      - However if you are looking at small cap growth stocks and they don't yet have that part of their portfolio filled they'll give you some more thought
- Good portfolio/profit

Hedge fund goals and metrics

- **Goals**
  - **Beat a benchmark**
    - Beat the SP500 index
      - It's OK if your fund goes down as long as the index goes down more
    - Pick an index you're an expert in
  - **Absolute return**
    - Must provide positive return no matter what
      - Slow gradual is fine as long as it's up
    - Long
      - Make positive bets in stocks that will go up
    - Short
      - Make negative bets in stocks that will go down
    - These types of funds don't make the same percentage gains as the beat a benchmark fund but they have very low drawdowns as in when the market takes a big hit, these funds usually don't
- **Metrics**
  - **Cumulative Return**
    - Given a start value, how much did I end up with after a certain period of time
  - **Volatility**
    - A measure of how rapidly and aggressively the portfolio goes up and down in value
      - Lower volatility is better
  - **Risk/Reward**
    - Sharpe Ratio - Risk Adjusted Reward: Reward / Risk



Computing inside a hedge fund

- Work backwards from the market
- We have a certain portfolio that is
    - Which stocks we have
        - And whether we're in positive or negative positions with regard to them
- The trading algorithm is simple
    - It's interacting with the market observing the live portfolio and what it's trying to do is get the live portfolio to match some target portfolio
    - Somewhere further back in the hedge fund we define a Target Portfolio
        - How many shares of Apple, how many shares of Delta Airlines and so on
        - The trading algorithm is trying to get us there
    - **So it's comparing target with live and then to move this portfolio towards that target it's issuing orders**
        - Orders like **buy** 200 shares of Apple (to the market)
            - Those orders get executed, or not and that updates the live portfolio
    - This kind of trading algorithm is important because you don't want to execute everything at once
        - Suppose you wanted to take a very very large position in Apple - if we were just to send an order of buy 10 million shares of Apple that could affect us detrimentally in the sense that the price for Apple would probably rise and we would not get a good **execution** so the algorithm must take these things into consideration as it moves our live portfolio to be more close to the target portfolio
            - Sometimes it takes days to get in a particular position so this doesn't happen all at once
        - There are many algorithms that have been invented to solve various problems
- You also have a N-day forecast of stock prices out to be which drives what our target portfolio out to be for today
- All this is fed into the **Portfolio Optimizer** that works the balance between risks and rewards for a balanced portfolio that considers volatility and correlation between different stocks and so on
- You also have historical data
    - Open, Close, Volume
    - You can look at this data to inform how stocks are correlated or uncorrelated to one another and also of **critical importance** is our current portfolio
        - It may be the case that if we're holding something, we don't want to exit it immediately because we'd be penalized by rapidly selling it
- The optimizer takes all this information into account to get to the target portfolio that our trading algorithm is working to drive us towards

How do we come up with the N-day forecast?

- **We have some kind of Forecasting Algorithm that takes in Historical Data and some form of Proprietary Information Feed**
- **This is the focus of machine learning!**

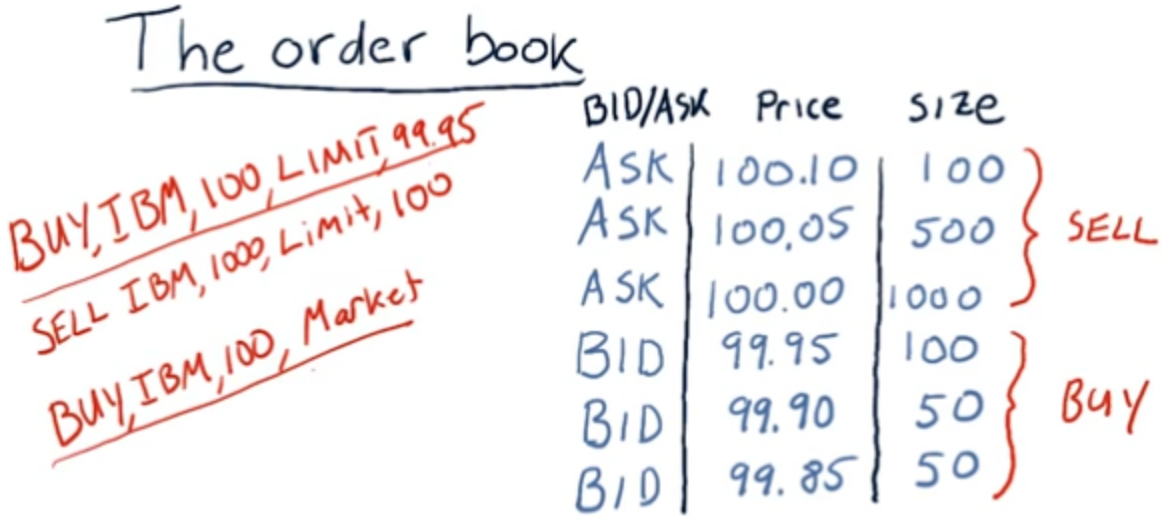
02-02 Market Mechanics

What is an order?

- **Buy** or **Sell**?
- Symbol
    - The identifier for the stock or ETF
- #shares - how many shares
- **Limit** or **Market**
    - **Market** means you're willing to accept a good price, but essentially whatever price the market is bearing
    - **Limit** means you don't want to do any worse than a certain price
        - If you're selling it means you specify you don't want to sell below a particular price
        - If you're buying it means you don't want to pay more than a certain amount to get it
- Price
    - You only specify the price if it's Limit
        - **e.g. BUY,IBM,100,LIMIT,99.95**
    - If it's Market you can't specify the price because you're getting whatever the market is at
        - **e.g. SELL,GOOG,150,MARKET**

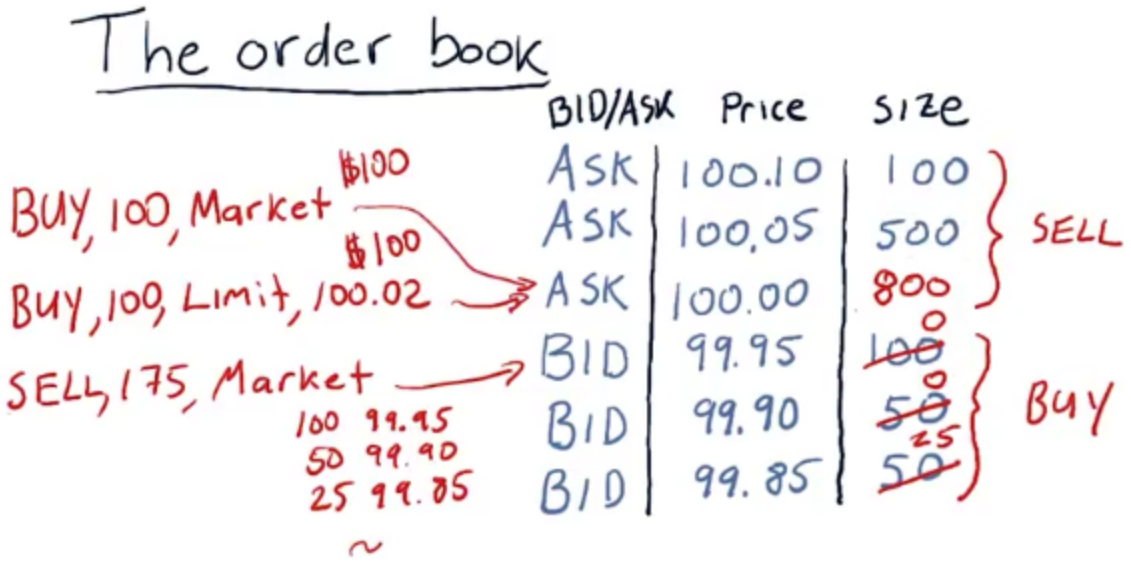
The **Order Book**

Each exchange keeps an order book for every stock that they buy or sell.
Order: BUY,IBM,100,LIMIT,99.95
In order book:
ASK 100.00 1000
BID,99.95 100 (or BID,GOOG,99.95 100?)



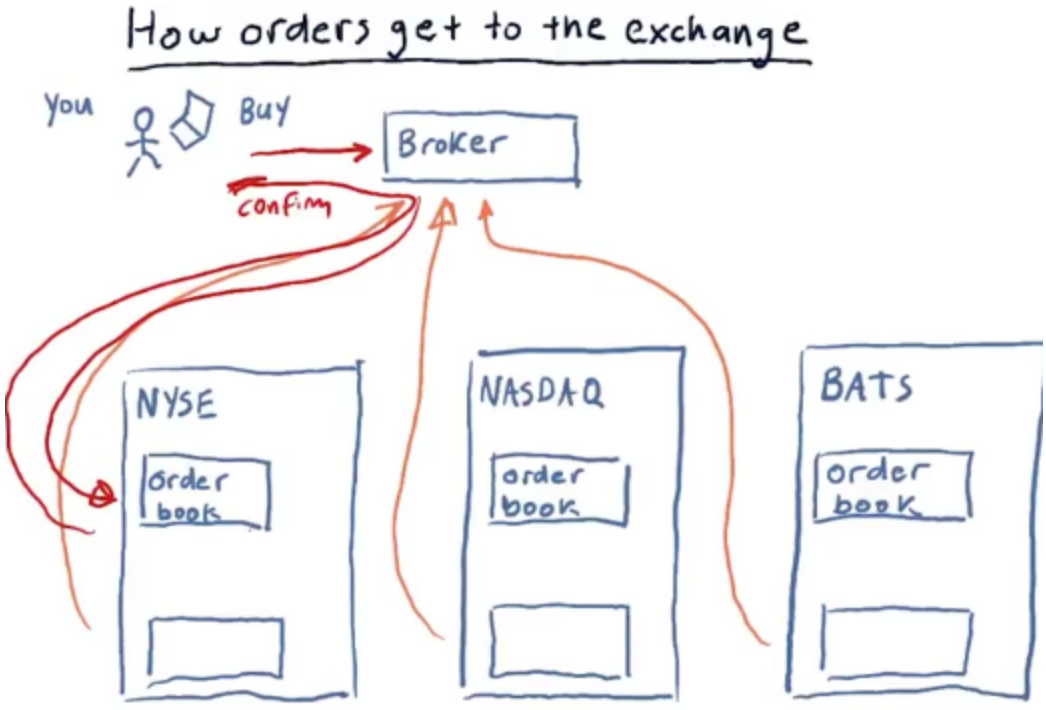
The price will go down because there are more shares that are being sold than people want to buy. There is much more selling pressure! If you put in a SELL order (at market value), you will sell the first 100 shares at $99.95, the next 50 at $99.90, and so on.In case of a BUY order, the market value won't change for the first 1000 shares. So, depending on your volume of trade, the value of the stock will likely go down.

The way to read that is that you don't make a buy of 500 at market price because the price doesn't budge, you make smaller volume buys like in the diagram in order to push the price down.



How orders get to the exchange

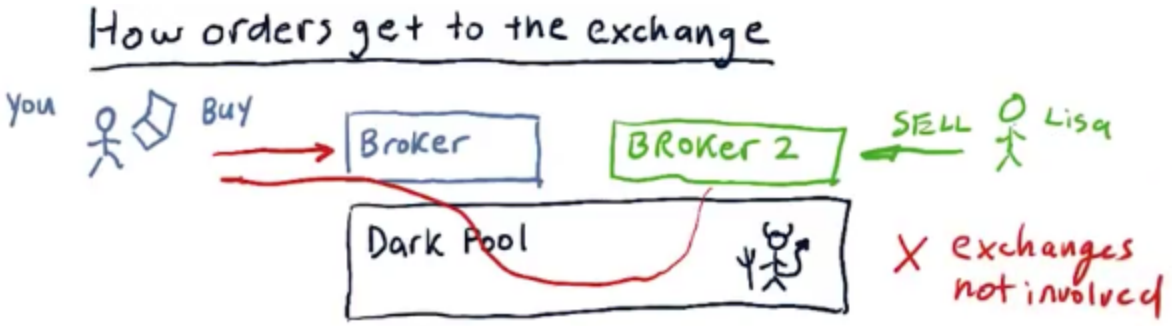
Scenario 1



Scenario 2

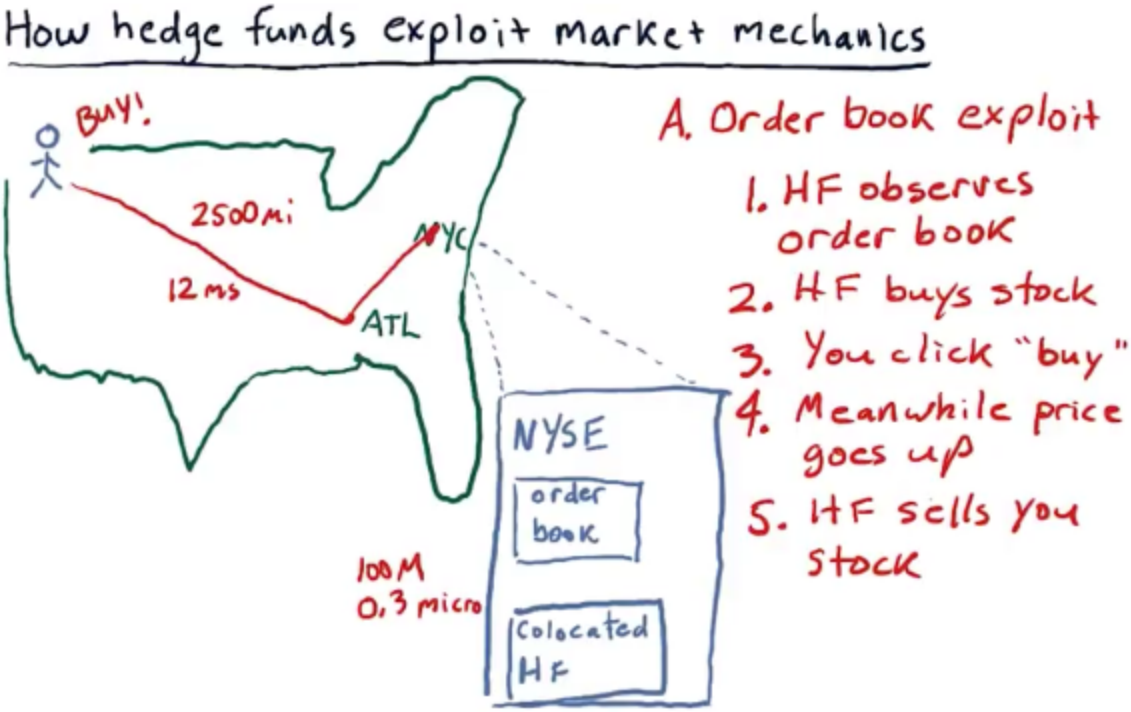
Scenario 3

Dark Pool



- Acts like an intermediary between brokerages and exchanges
- The dark pool is looking at the order books of the various exchanges and they're often making predictions about which direction stocks are going to go
    - They pay the brokers for the privilege to look at the orders before they go to the exchanges and if they see an advantageous trade they'll go ahead and take it
- In fact these days 80-90% of what they call retail orders never make it to the exchanges

How hedge funds exploit market mechanics



1. Hedge Fund observes the order book at 0.3 microseconds
2. HF buys stock
3. You click "buy" and it takes 12+ ms for the order to come in
4. Meanwhile the price goes up
5. HF sells you the stock while essentially skimming off the top
    - They're exploiting the remote people from around the country because they can act much earlier
    - They don't hold on to the stock for more than a few milliseconds

This rarely happens and it's usually by fractions of a penny but the hedge funds will be there to pick those pennies off the ground

Additional order types

Exchanges

- **Buy, Sell**
- **Market, Limit**

Broker

- **Stop Loss**
  - When the stock drops to a certain price, sell
- **Stop Gain**
  - When the stock reaches a higher price sell
- **Trailing Stop**
  - Is a combination of stop loss but also an automatically changing value for when that criteria is met
    - For instance, you might have this trailing stop remain say, %0.10 behind the price, so as the price goes up, the value at which you would want to sell the stock goes up along with the price, but when it drops down below then that stop loss is triggered
- **Selling short, the most important and the most impactful kind of order**
  - **Allows you to take a negative position on a stock. In other words, you sell a stock short if you believe its price is going to go down. Keep in mind, we're selling stock we don't even own**

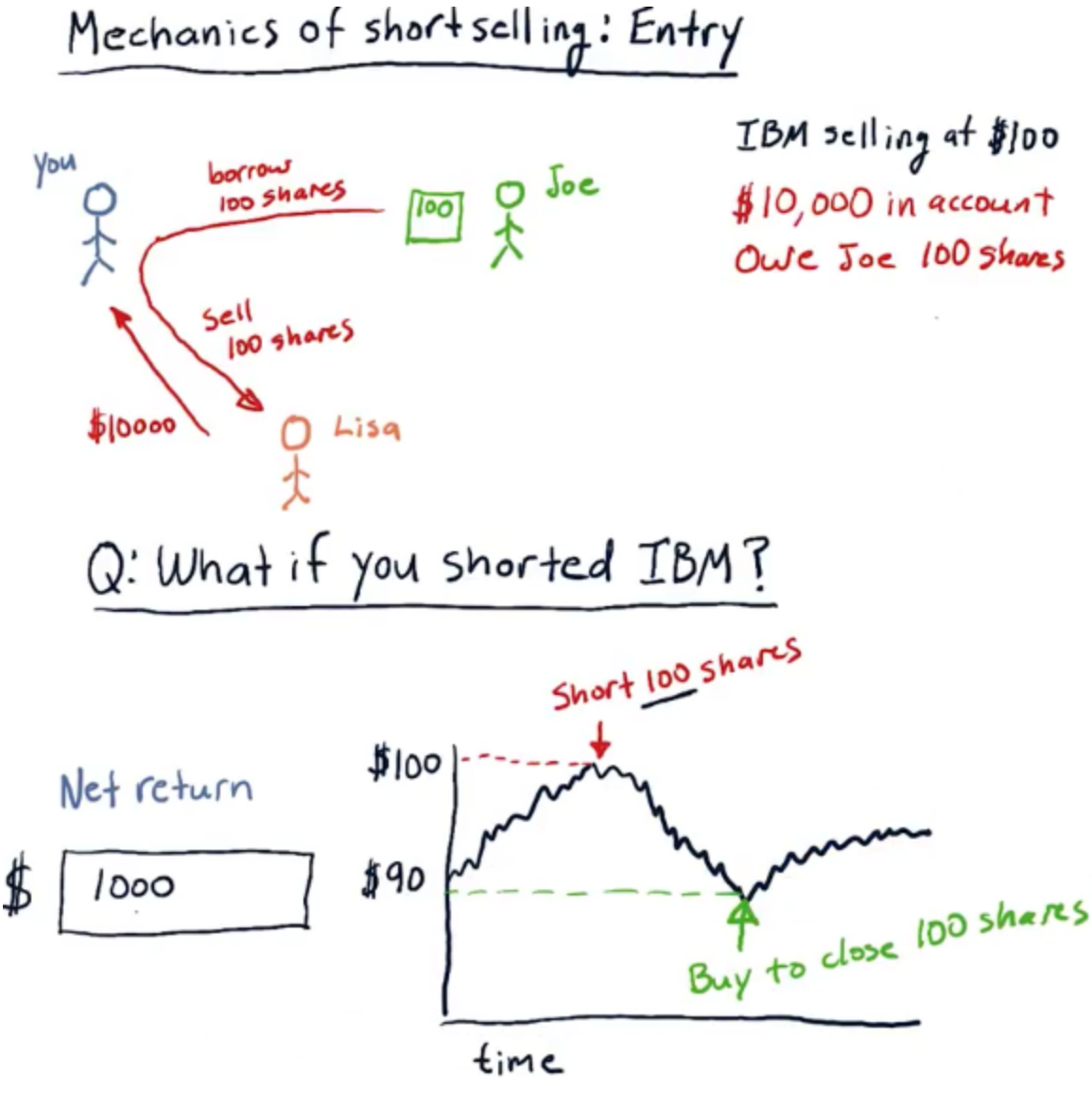
Mechanics of short selling: Entry

- You want to take a short position in IBM and IBM is currently selling at $100 that is the current market price
- Joe holds 100 shares of IBM - he likes IBM he wants to hold on to it - but he's willing to lend you those shares of IBM (Joe's broker will take care of that for him). Joe may not even know he's lending you the shares.
- Lisa thinks that IBM is going to go up and she wants to buy IBM

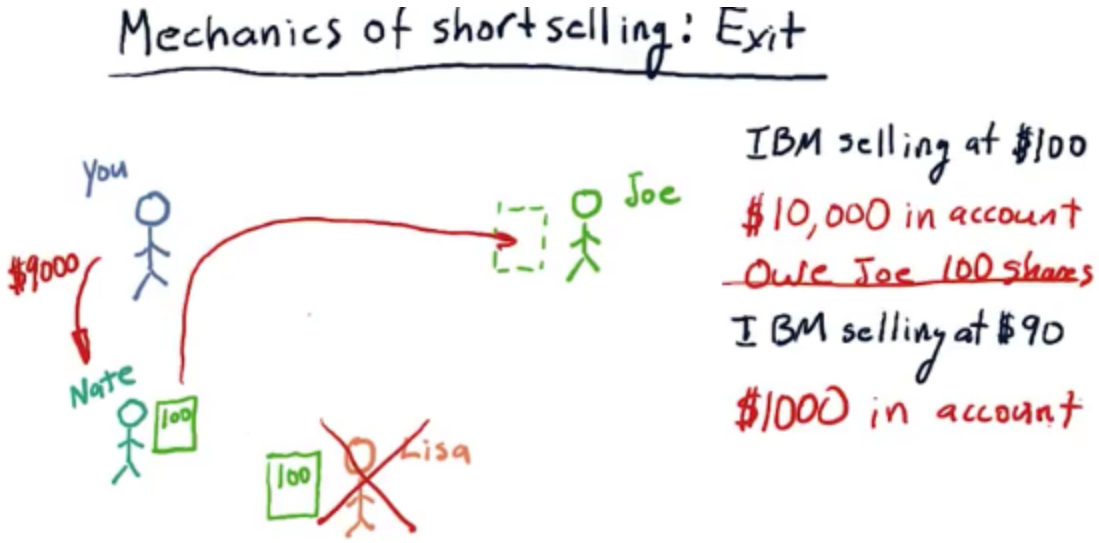
So far you want to sell IBM short, and you don't own any shares of it, Lisa wants to buy IBM, she thinks it's going to go up

So here is what happens:

- You borrow that 100 shares from Joe
- Now that you have those shares you can turn around and sell them to Lisa
- In exchange for those 100 shares that you gave Lisa she gives you 100 times $100, or $10,000
- Now we have $10,000 in account and we owe Joe 100 shares of stock
- The problem that arises here is that Joe may decide he wants his 100 shares back. What you'll have to do is go buy them from someone and then give them back to Joe



Lisa may say no for selling you back the IBM she bought, but Nate may be willing to sell.



You don't actually make these personal agreements yourself, the broker does.

What can go wrong?

Let's go back to where we had $10,000 in our account and we owed Joe 100 shares of stock, what can go wrong?

- What if instead of going down, IBM went up to $150 per share and now you want to exit your position?
- You again go to Lisa and say hey Lisa do you want to sell me your shares of IBM, she says no but Nate is still out there
- But this time the price is $15,000 because the price has gone up and the net result is -$5000. So if you bet wrong when you short a stock and the price goes up you lose money and it can be significant

02-03 What is a company worth?

If it provides $1/year reliably, guaranteed.

- If the company produces $1 the first year, and sticks around to produce one more the next, it is already worth more than $1, isn't it? Of course, there's a chance that the company goes belly up!
- If you live exactly 70 more years, it might seem logical to pick $70 - this is how much the company is worth to you. But you're assuming that the value of $1 produced each year will remain the same. How does the value of currency typically change over time?
- This is a close one - the value of the company may seem to be infinite as it will keep generating $1 every year! But you're assuming that the value of $1 produced each year will remain the same (or grow). How does the value of currency typically change over time?

The most correct Answer: **It's worth between $10 and $25 depending on interest rates**

- The value of $1 would typically reduce over time, and the sum of the individual $1's generated every year would converge to a finite value.

Why company value matters

We want to buy a stock when the price is too low and we want sell when it's too high.

- **Intrinsic value**
  - This is based on the value of the company as it's **estimated by future dividends**. Many companies, not all, pay each year or each quarter a **dividend - a cash payment**, if you own a share stock you get a certain amount of dividend
  - So **Intrinsic value** is based on if we own one share stock, we're going to get some amount of dividends accumulated over all of the future - what is the company value based on that?
  - If I own a share of stock how much money am I going to get over the future and what's that worth overall in the future going forward?
- **Book value**
  - Based on the assets the company owns - so in other words we add up the value of all the factories that they own and so on
  - **This relates to what's the value of the sum of the assets?**
- **Market cap**
  - This is based on the value of the stock of the market and how many shares are outstanding
  - **So this is based on what does the market think the company is worth**

The Balch Bond

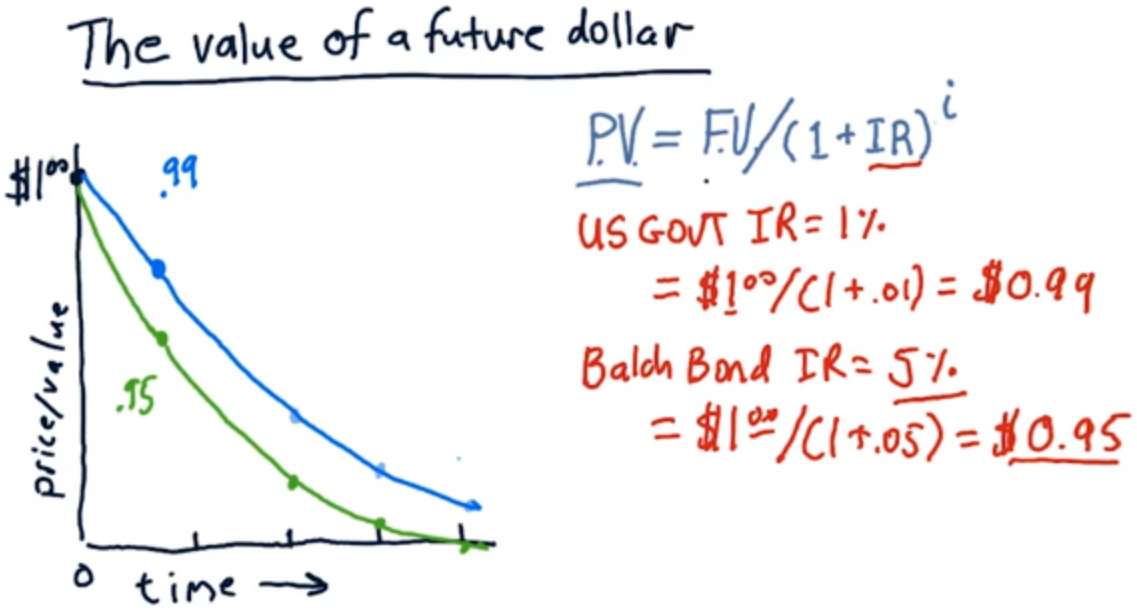
- We're going to consider now the intrinsic value of our $1 machine. In other words this machine generates one dollar every year. To understand that we need to understand what is the future value of a dollar
- Suppose I told you - hey look, awesome student, I promise I will give you one dollar in 1 year, how much would you pay me today on the basis on that promise? In other words, maybe you would pay me 80 cents and then in a year I give you $1, maybe you would pay me more

Intrinsic value: The value of a future dollar

- **It should be obvious now that the value of a dollar delivered in the future, even if it comes from the US government is worth less than a real dollar right now.**
- **A real dollar right now is worth $1. If I promise to give you a dollar in the future, it's not worth as much.**

The reason is that there is a chance that the entity promising the $1 will not deliver on that guarantee. The US Govt is more likely than Balch to deliver on it though. This amounts to risk and interest rate.

- P.V. = F.V. / (1 + I.R.)^i
  - Present Value
  - Future Value
  - Interest Rate
  - The i is how far into the future this payment is going to be delivered, in years



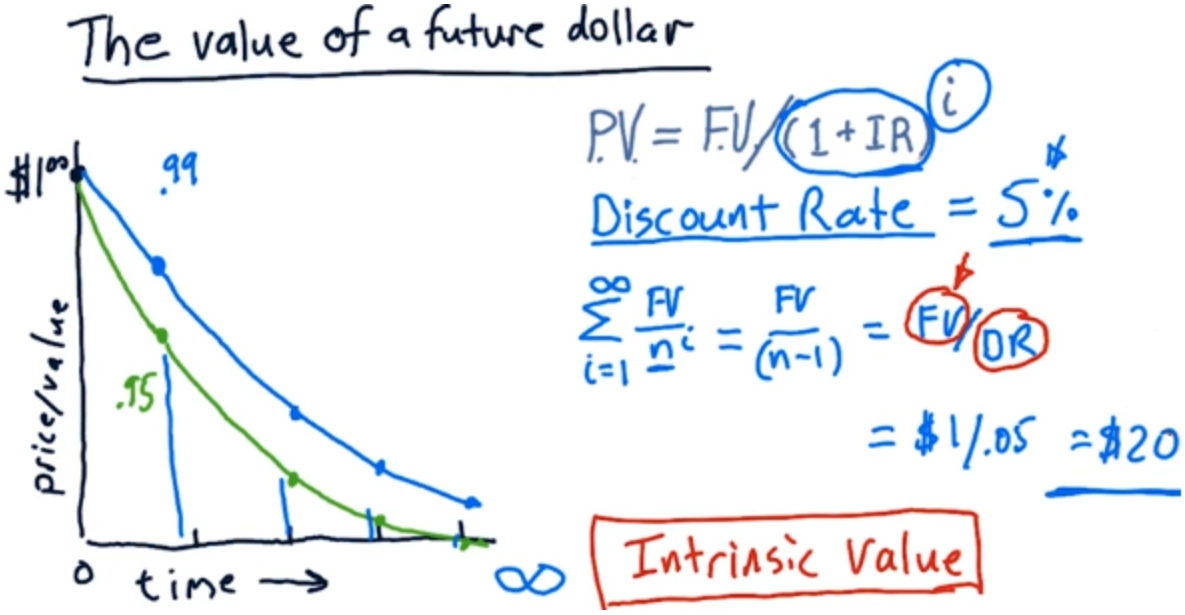
What is the Present Value of a Future Dollar delivered some number of years into the future where i is how many years it is

***Interest Rate* and *Discount Rate* are terms that refer to essentially the same quantity, but are used to distinguish two slightly different use cases:**

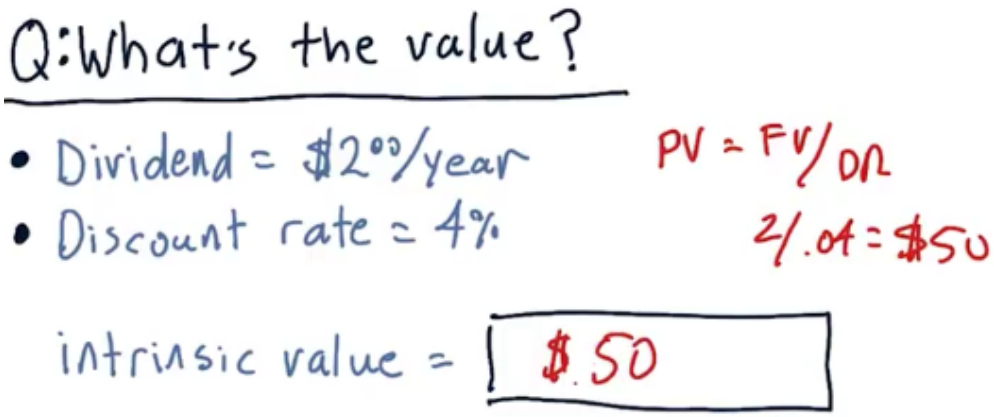
- **Interest Rate is used with a given Present Value, to figure out what the Future Value would be.**
- **Discount Rate is used when we have a known or desired Future Value, and want to compute the corresponding Present Value.**

For instance, in this case we want to sum up all future dividends - equal to a constant ($1 or FV) every year.

- The Interest Rate I.R. relates to how risky the company is - in other words if you're as sure that it's going to pay you that dollar every year as you are sure that the US government would pay you a dollar on a bond, then the IR should be the same as the US Govt bond rate. However if you're less certain that this company is going to pay you at that rate, this interest rate needs to be a little bit higher.
- You need to expect that the company is going to pay you more in the future or that it's going to be more likely to pay you that dollar in the future - this is called the **discount rate.**
    - **The discount rate is higher if you trust the company less or you think it's more risky**
    - **The discount rate is lower if it's more certain to pay you that dividend every year**



- **FV future value**
- **DR discount rate**
- **N = (1 + IR)**
- **The value of the dollar machine is $20 overall**



Book Value

"Total assets of the company minus intangible assets and liabilities"

- Assets are things like property that is owned
- Intangible assets
    - Things that are difficult to put a price on like the value of a brand or patent
- Liabilities
    - Loans that are owed

❖ Let's say that the company owns 4 factories and each factory is worth $10M

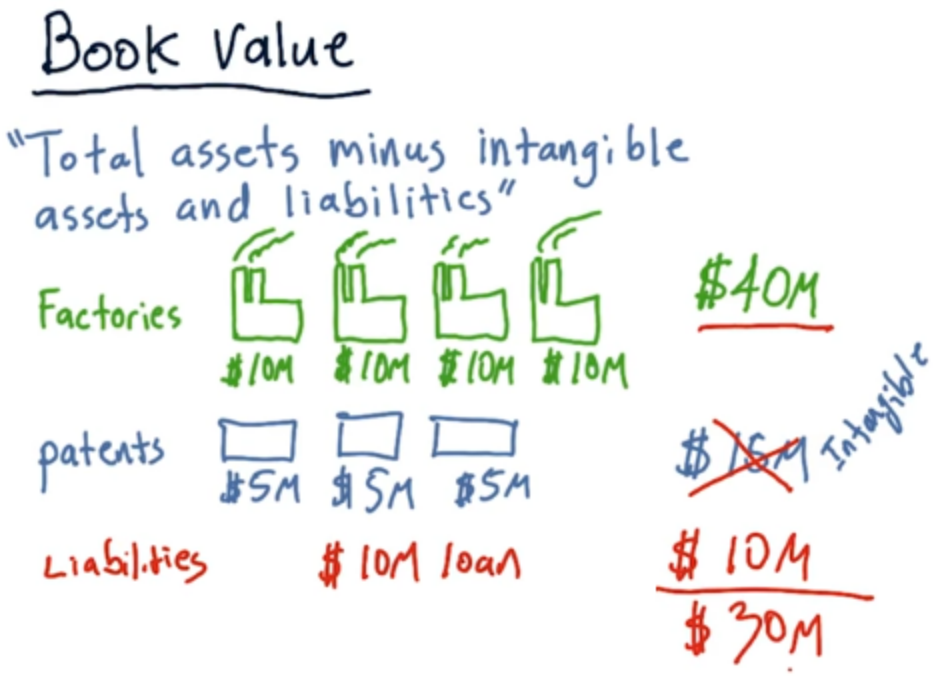
➢ Total 40M

❖ Also holds three very important patents and each is valued $5M

➢ Total 15M (these are not counted)

❖ Liabilities $10M loan

Final Total book value: $40M - $10M = $30M



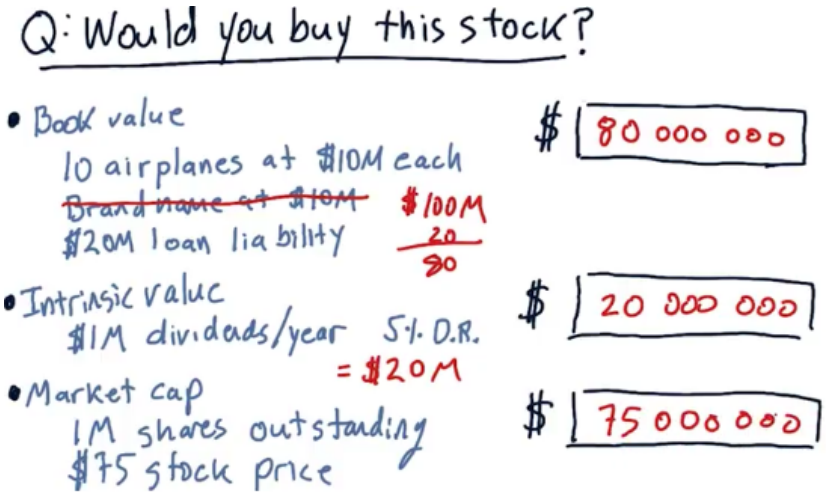
Market capitalization

Market cap = #shares * price

Why information affects stock price

News influence the expectation of future dividends, the intrinsic value and why people may want to play less or more for that stock. They reduce or increase it.

- **CEO ill**
  - Is company specific news that affects the price of an individual company
- **Island soil is Bad**
  - Sector specific news that affect companies on a particular island or in a particular industrial segment
- **Sea level rising**
  - Market wide news that affects all the companies in the market



Yes, you should buy it right away!

Ignoring the intrinsic value, if you buy the entire company off the market (for $75M) and immediately sell it for its book value ($80M), you have a $5M profit right there!Even if you are buying some stocks (instead of the whole company), the stock price is expected to increase (as it is currently undervalued).

This is why Stocks very very rarely go below their book value - because if they ever go below their book value a predatory buyer will buy the entire company and then break it up and sell it's pieces.

Summary

We talked about three key ways to compute the value of a company:

- **Intrinsic Value**
    - Which is based on future dividends - companies pay a certain amount every year based on how many shares they own and this is the value of all future dividends going into the future
- **Book Value**
    - This is the value of the company if we split it up into pieces and sold those individual pieces
- **Market Cap (Capitalization)**
    - Which is simply the value the market is placing on the company

Many stock trading strategies look for deviations between the intrinsic value and market capitalization

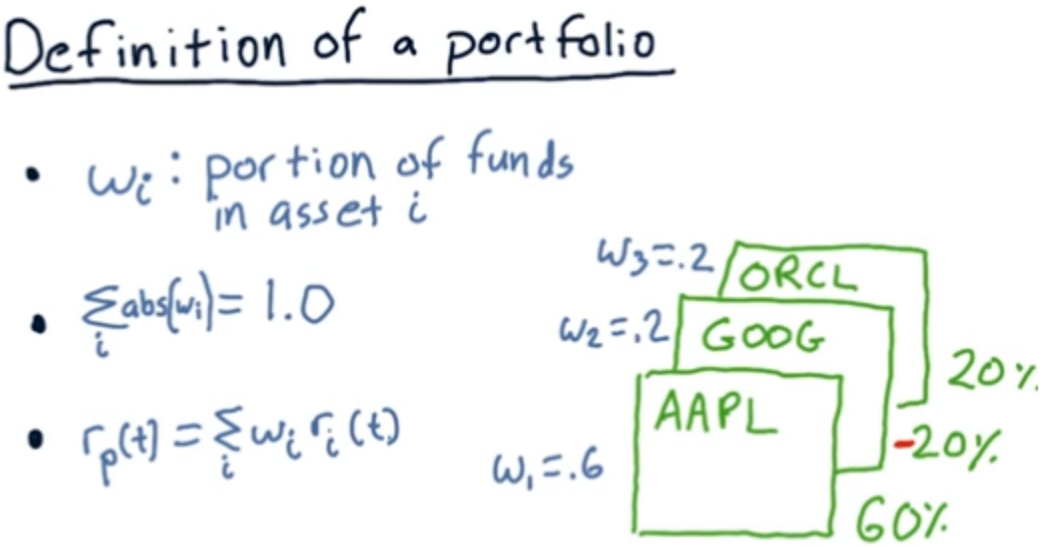
- If the intrinsic value drops significantly and the stock price is high, it may be worthwhile to short that stock
- If dividends are going up and the market cap is low, it is an opportunity to buy the stock
- Similarly the book value provides a sort of lowest price when stock price begins to approach book value you can pretty much assume that the price is not going to go below book value, not much below it because otherwise a predatory buyer will buy the company and break it apart

02-04 The Capital Assets Pricing Model (CAPM)

- One of the most significant ideas affecting finance in the last century
- It explains how the market impacts individual stock prices
- It provides a mathematical framework for hedge fund investing
- There's a nobel prize involved
- There's a belief that you can't beat the market

Definition of a portfolio

A portfolio is a weighted set of assets.



- We use abs(wi) because **negative allocations are short**
- **rp** are returns on a portfolio

QUIZ:
Assume
Stock A +1%
Stock B -2%
Wa = 75%
Wb= -25%
0.75 * 1% + (-0.25) * (-2%)
0.75% + 0.5%
= 1.25%

The market portfolio

When someone refers to the Market, they're usually referring to an index that broadly covers a large set of stocks. In US the best example of that is SP500. The SP500 represents the 500 largest companies that are traded on exchanges and that index changes each day according to the prices of all its components.

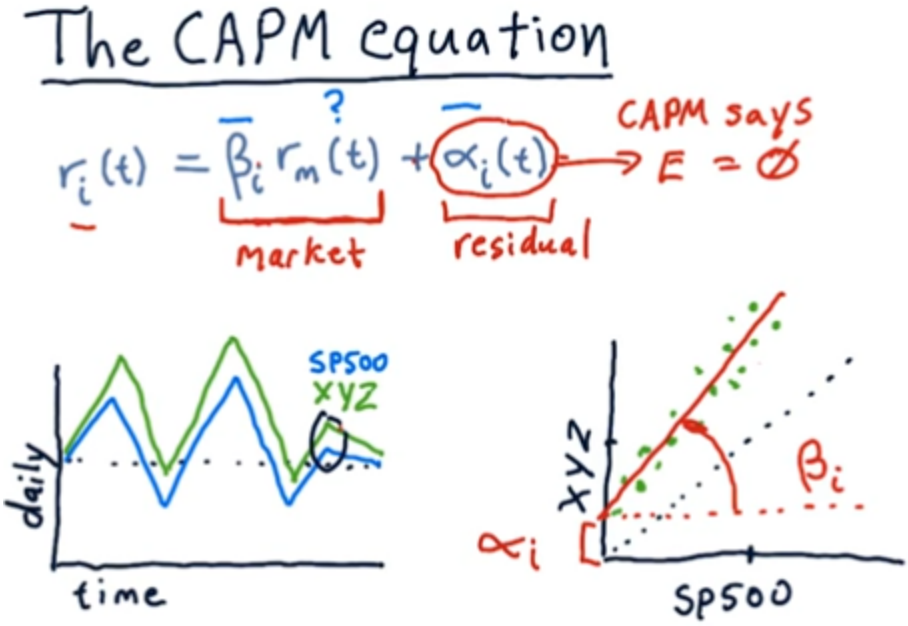
Each index is Cap Weighted

You take the market cap and divide by the market sum of all the stocks

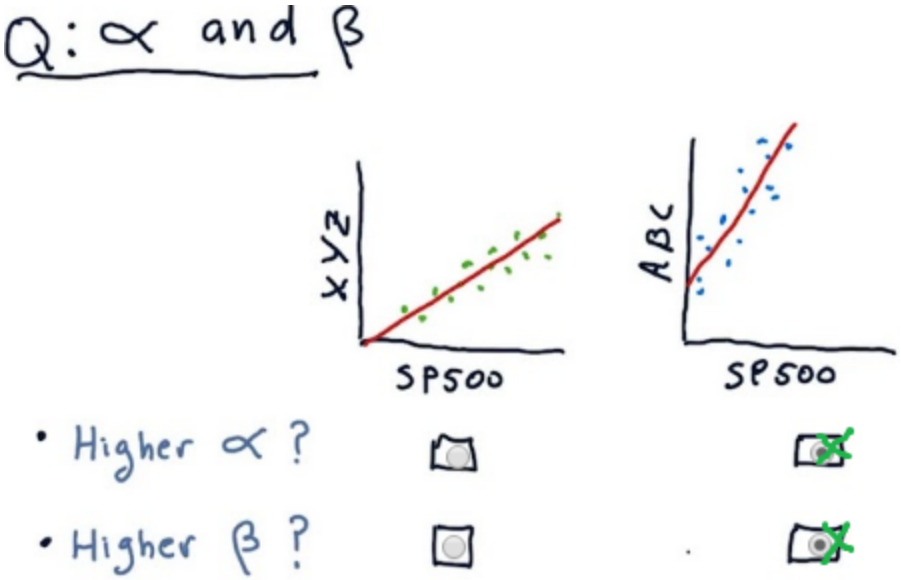
The CAPM Equation - must memorize

$$r_i(t) = \beta_i r_m(t) + \alpha_i(t)$$

- **r** is return
- **i** is a particular stock
- **ri(t)** is return of a particular stock on a particular day
- m is market
- rm(t) is return on the market for that day
    -
- The capital assets pricing model says that significant portion of the return for an individual stock on a particular day is highly dependent on the market
- **Beta** is the extent to which the **market** affects a particular stock, and it's also the market
- **Alpha** is the **residual** - so if you look at all the stocks over one day and look at how much the market goes up or down and you calculate how much the stock should have gone up or down according to **Beta**, there'll be something left over. It won't exactly match what beta predicted and that's what this alpha or residual component is



CAPM says that the expectation for Alpha is always 0: E = 0
Beta and **Alpha** come from daily returns and essentially how the daily returns for a particular stock relate to the daily returns of the market.



Correct!
Correlated with SP500, ABC clearly has a greater slope than XYZ, therefore higher β.

It also has a larger *Y-intercept* (α).

Correlated with SP500, ABC clearly has a greater slope than XYZ, therefore higher β. It also has a larger *Y-intercept* (α).

CAPM vs. Active management (Passive Investing vs. Active Investing)

- **Passive**: buy index and hold
- **Active**: pick stocks
    - Overweight (higher weights for some stocks)
    - Underweight (lower weights for other stocks)
    - Weights could be 0 or even negative (shorts)
    - The general idea here is that the active portfolio manager's portfolio differs from the market portfolio by selecting different weights on different stocks

- Both active and passive managers agree with $\beta_i r_m(t)$ as in how the stock moves each day is most significantly influenced by the market and that the amount that it moves is strongly related to beta
- Where they differ is $\alpha_i(t)$ is with regard to their treatment of **Alpha**
    - **The CAPM (Capital Asset Pricing Model)** says two important things about alpha
        - Passive Managers
            - It's random, you can't predict it
            - The expected value is zero, E(Alpha) = 0
                - Some days it will be positive, some day it will be negative
                - It may be large, it may be small, but it's random and on average the value will be zero
        - Active Managers
            - Believe they can predict Alpha
                - They can compare two stocks and they say: "I think this one is going to go up or down relative to the market"
            - They may not be exactly right on every single pick but they believe on average they're better than just flipping a coin
            - If you believe active managers, you can use information and machine learning to find stocks that have either positive or negative alpha and you can use that information to select your stock picks

- **If you believe the CAPM then you should be a passive investor - just buy an index and hold it**
- **If you believe active managers that they can find alpha then you should consider being an active investor**

CAPM for Portfolios

$$r_p(t) = \sum_i w_i(\beta_i r_m(t) + \alpha_i(t))$$

$\beta_p = \sum_i w_i \beta_i$:: **this is the beta for the entire portfolio**

$$r_p(t) = \beta_p r_m(t) + \alpha_p(t)::$$

- So CAPM (the capital asset pricing model) tells us that we can simplify the expected overall return for a portfolio by computing a beta for that portfolio, multiplying it by the return on the market and then there's some **Alpha_p**.
- That **Alpha_p** is composed of multiple examples of **Alpha_i**, but we don't need to bother to add them up (because on average they're 0s) and we can approximate by an overall portfolio alpha
    - ■
- Passive Manager accepts that this is what CAPM says that we can expect on a particular day and again CAPM asserts that this alpha is random with mean 0.
- Active Manager will say yes, I agree that the beta component is the same as CAPM says, but I think I can find value in this **alpha** and it needs to be broken out individually
    - $r_p(t) = \beta_p r_m(t) + \sum_i w_i \alpha_i(t)$

Implications of CAPM

- In Upward Markets
    - **You want a higher β in upward markets so that you can ride the surge**
- In Downward Markets
    - **But a lower β in downward markets so you don't crash as much**
- The expected return on our portfolio, is the beta of the portfolio times the return on the market plus alpha of the portfolio $r_p = \beta_p r_m + \alpha_p$
    - The first important point is that alpha is random and the expected value of alpha is 0
        - This excludes alpha from our toolbox of ways to make money
    - Only way to beat markets is by cleverly choosing **Beta**
        - **Choose high Beta in up markets**
        - **Choose low Beta in down markets**
        - The only problem with this is the Efficient Market Hypothesis (EMH) which says that you can't predict the Market so these are not available to you
- **CAPM says you can't beat the market**
- **If CAPM is true, we can't beat the market**

- **We don't believe that and this whole course is about ways that you can potentially beat the market - so we're getting to these eventually but CAPM is a reference framework for all the other things**

Arbitrage Pricing Theory (APT)

- Not used in this class
- Stephen Ross 1976
- $r_i = \beta_i r_m + \alpha_i$ this only allows us to consider a stock in the entire ocean
    - Stephen Ross's idea was that you have to consider the different islands
        - In other words, a particular stock might have exposure to different aspects of the market
            - Exposure to finance $\beta_{iF} r_F$
            - Exposure to Tech $\beta_{iT} r_T$
            - Exposure to Mfg $\beta_{iM} r_M$
        - We can compute for each stock, for each sector an individual beta
    - His assertion was that by breaking out the betas into the different factors we could get a more accurate forecast of what the return's going to look like:
        - $r_i = \beta_{iF} r_F + \beta_{iT} r_T + \beta_{iM} r_M + \alpha_i$

CAPM == OCEAN
APT == ISLANDs in the OCEAN

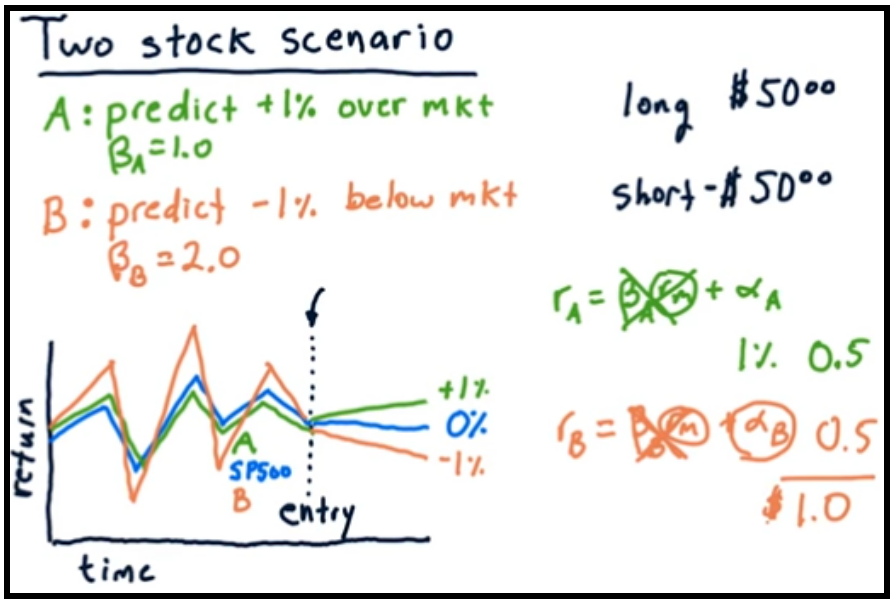
02-05 How hedge funds use CAPM

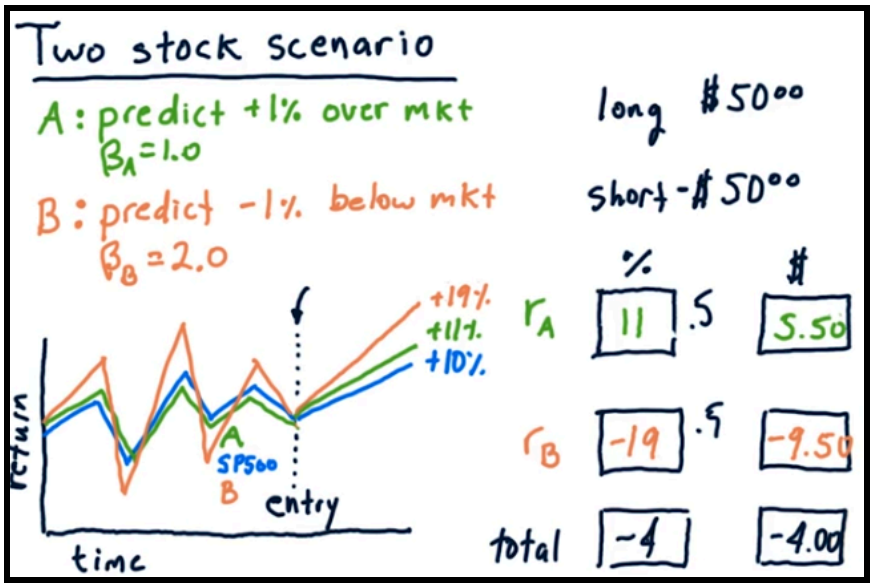
- A typical hedge fund develops methods to find stocks they think will perform well
- The informational edge that they're seeking is usually market relative
    - This means that they're looking for stocks that will go up more than the market if the market goes up
    - Or stocks that will go down less than the market if the market goes down
- If this information is reliable, they can take advantage of the CAPM to virtually guarantee positive return

Two stock scenario

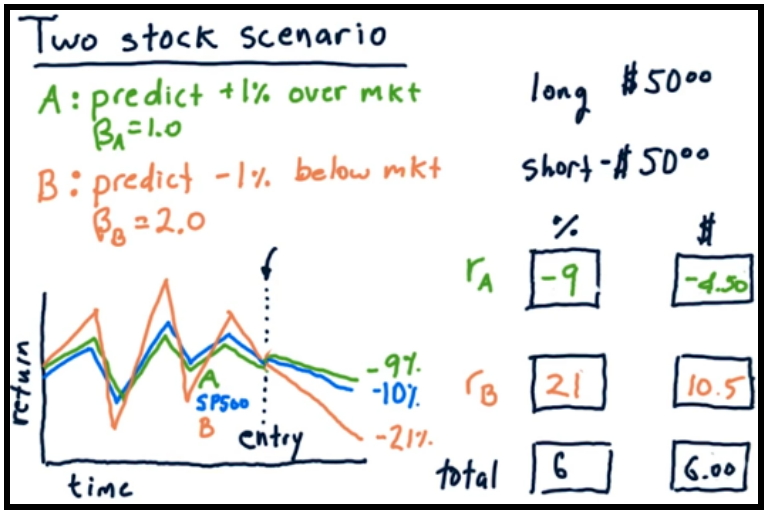
The **long short portfolio** - we think that stock A's going to go up so we should long it, we think that stock B's going to go down so we short it.

- There are some strong advantages to this long short approach that'll begin to emerge



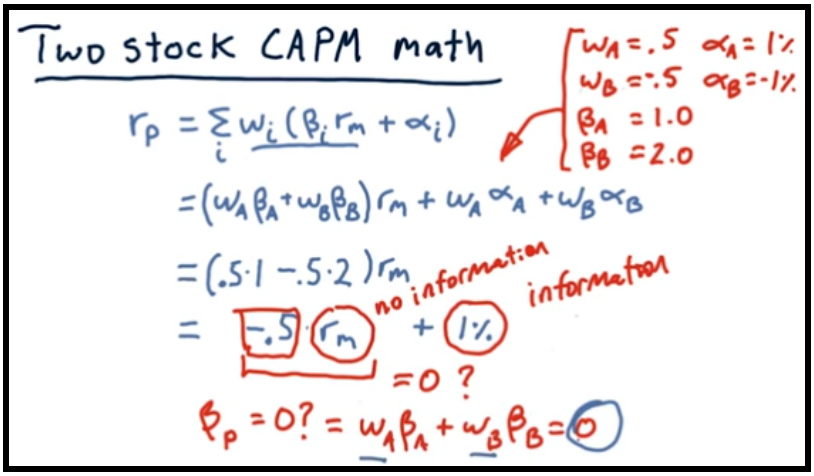


- You can't just add ra and rb to get -8, you need to take the allocation into account: ra*.5+rb*.5 = -4

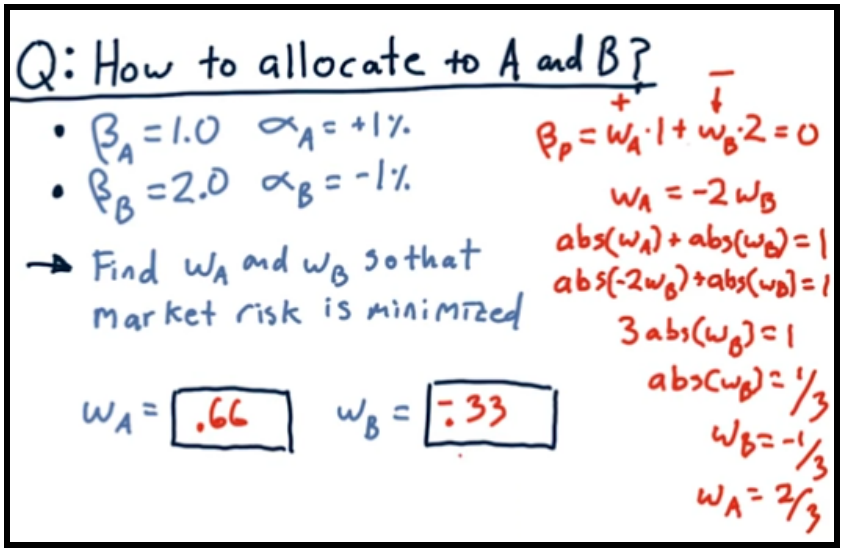


- **The takeaway is that even if we have perfect beta and perfect alpha, and we're not careful how we allocate our money, we can still lose**
    - How can we fix this? CAPM can help

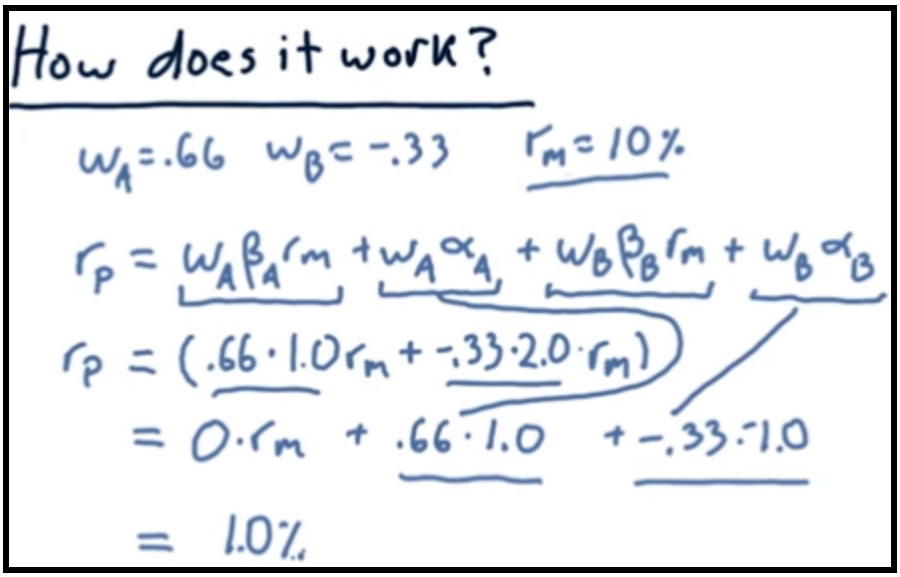
Two stock CAPM math



- We want to make the no information component 0 in order to eliminate market risk from our portfolio
- What are the weights wa and wb that will allow us to make that happen?



- Take a close look at Beta_p

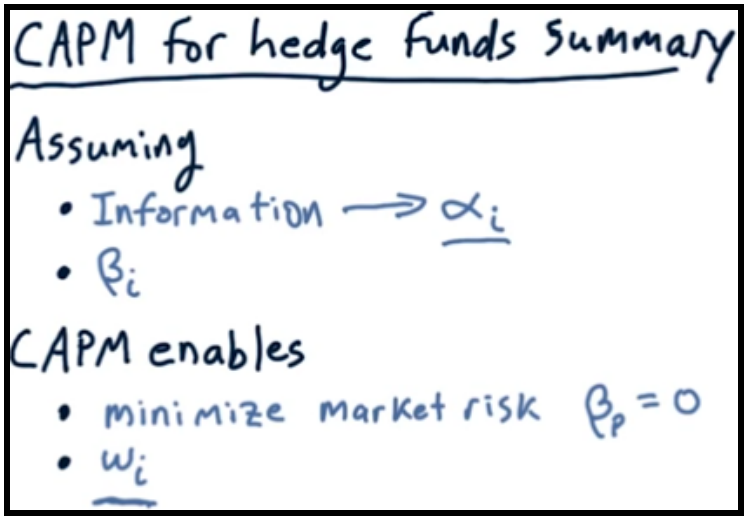


Caveats

- These betas aren't necessarily fully guaranteed to continue into the future
- These alphas aren't necessarily fully guaranteed to continue into the future
- These betas and alphas are just estimates that we computed based on information we thought we had
- It's not guaranteed by any means but it is a way to use long/short investing to reduce exposure to the market overall and to **focus on those alpha components where we do have information**

CAPM for hedge funds summary

- Assuming
    - We have some sort of actionable information that we convert into a forecast alpha, in other words a prediction for a particular stock i about which way it's going to go
    - Also the beta for that stock with regard to the market
- CAPM enables
    - Minimize market risk by finding a Beta_p = 0
        - And we can do that by finding the appropriate weights w_i on each individual stock i
    -
- This is where the whole idea of long short trading came about and CAPM really enables you to refine that



02-06 Technical Analysis

- There are two broad categories of approaches to use for choosing stocks to buy or sell. They're based on:
    - **Fundamental Analysis**
        - It involves looking at aspects of a company in order to estimate its value
        - Fundamental investors typically look for situations where the price of a company is below its value
    - **Technical Analysis**
        - Technicians don't care about the value of a company, instead they look for patterns or trends in a stock's price

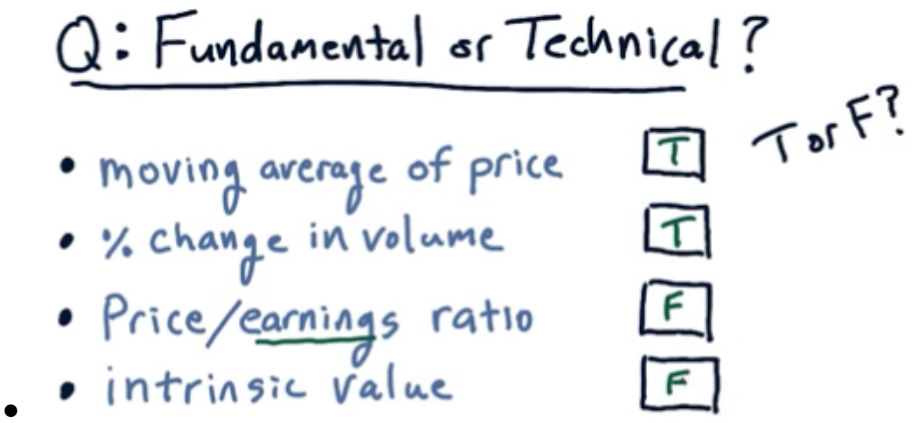
Characteristics of Technical Analysis

What is it?

- Historical price and volume only
    - It only looks at price and volume
        - (Fundamental analysis looks at fundamental factors like earnings, dividends, cash flow, book value and so on)
- Compute statistics called indicators
    - Indicators are heuristics that may hint at a buy or sell opportunity
- There is significant criticism of the technical approach
    - Folks consider that it's not an appropriate method for investing because it's not considering the value of the companies
        - Instead maybe you could think about it as a trading approach as opposed to an investing approach
- Still, there are reasons to believe that a technical analysis might have value and that it might work

Why it might work?

- There is information in price and information in price change
- Heuristics work
    - It reflects sentiments of buyers and sellers and especially if we see that the price for a particular stock is moving in a different direction than the overall market, that might be a hint that there's something going on
- Additionally we know that in other domains of artificial intelligence, heuristics can work and they work frequently
- Even though it's controversial, there's reasons to believe that technical analysis can work



When is Technical Analysis effective?

- **Individual indicators** by themselves are **weak**ly predictive
    - Back in the 80s and 90s when some of these were created - they had stronger value - but since that time more and more people have been trading according to them and essentially the more people who are following a particular approach, the less value is realized by any person by themselves - so individual indicators are weak
- **However** combining multiple indicators adds value. **Combinations are stronger**
    - Combining 3 to 5 indicators into a machine learning context provide a much stronger predictive system than just an individual indicator
- **Look for contrasts (stock vs. market)**
    - In other words look for one stock that has a strongly different indicator than another stock that is contrasting to the market
    - If all stocks are behaving in the same way as the market there's no reason to pick any one stock over another, but if you see certain stocks are behaving differently than the market, then they are worth a further look
- **Shorter time periods**
    - Generally works better over shorter time periods than longer time periods

When does Technical Analysis have value?

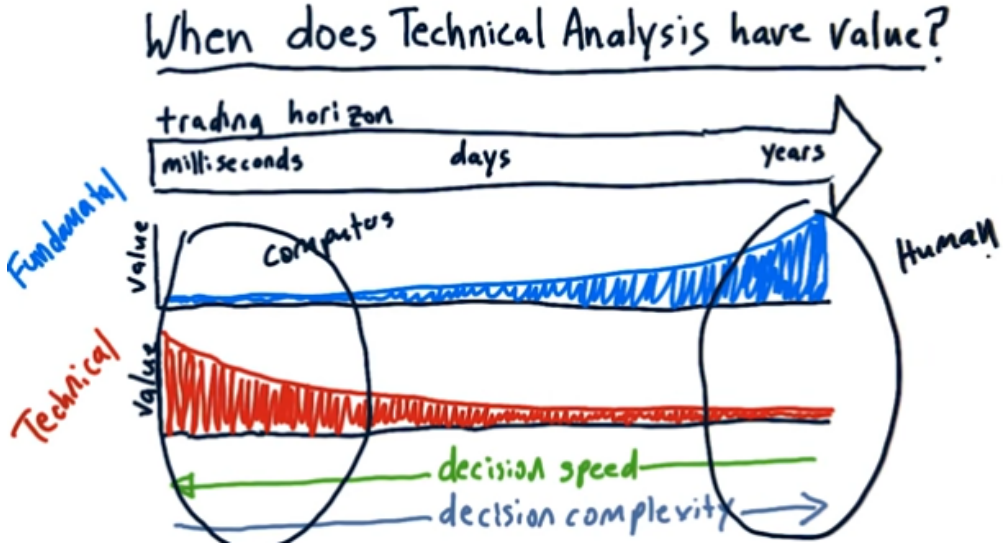
Consider the Trading Horizon

- When you buy a stock and then sell it, what's the time period between those two activities
    - It can be ms, days or years

For a long period of time, Fundamental Factors provide a lot of value
Over long terms, Technical Analysis is not so valuable

Consider:

- Decision speed - how fast must the decisions be made?
- Decision complexity - how complex is the decision to buy or sell stock if you're going to hold it for years



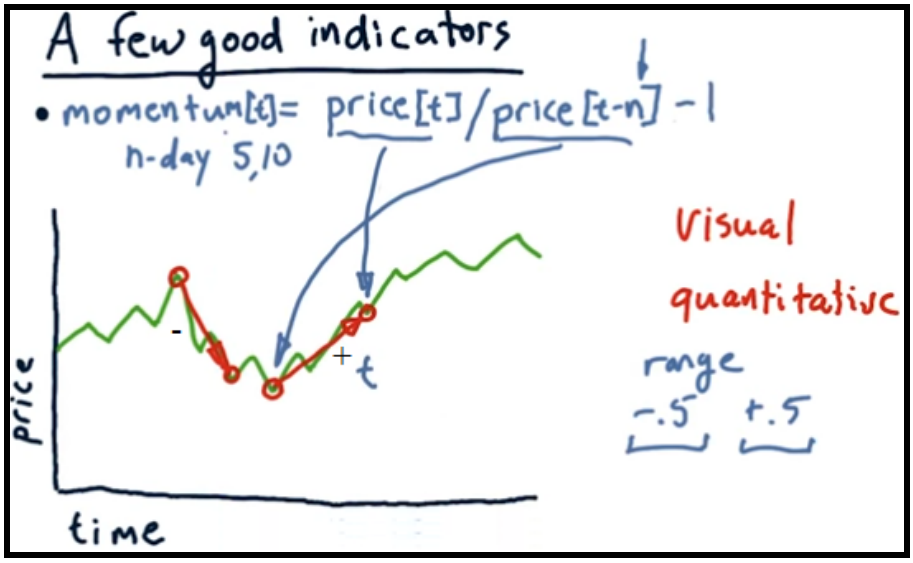
A few good indicators

There are hundreds of technical indicators out there but we're only looking at 3

- Momentum
- Simple moving average (SMA)
- Bollinger™ Bands

Momentum

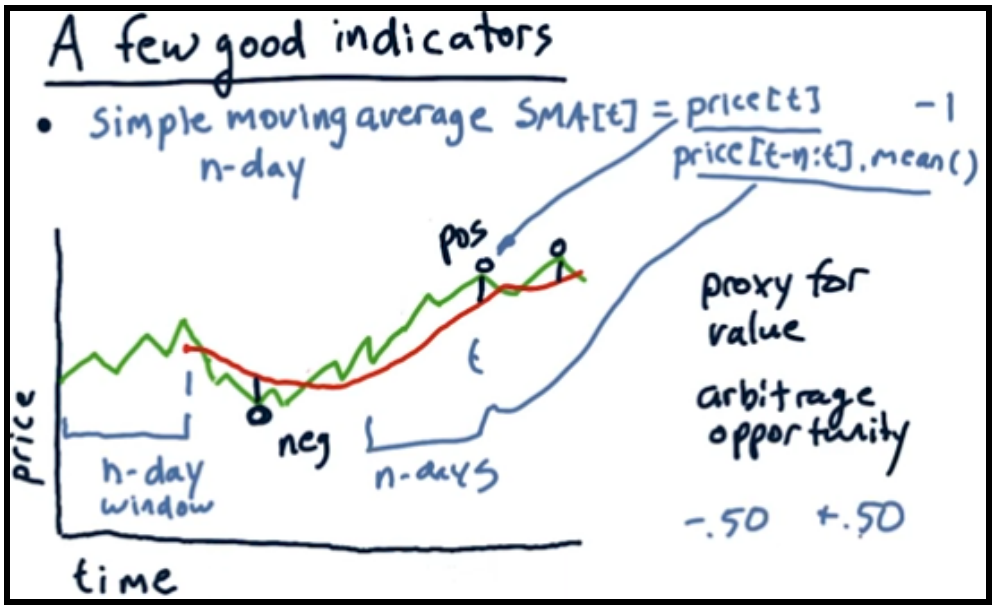
Over a number of days, how much has the price changed?



Simple Moving Average (SMA)

Also indexed by n, over how many days we're looking back: n-day window.
It's important to notice that it lags the movement
There are at least two different ways technicians use SMA as part of trading strategies:

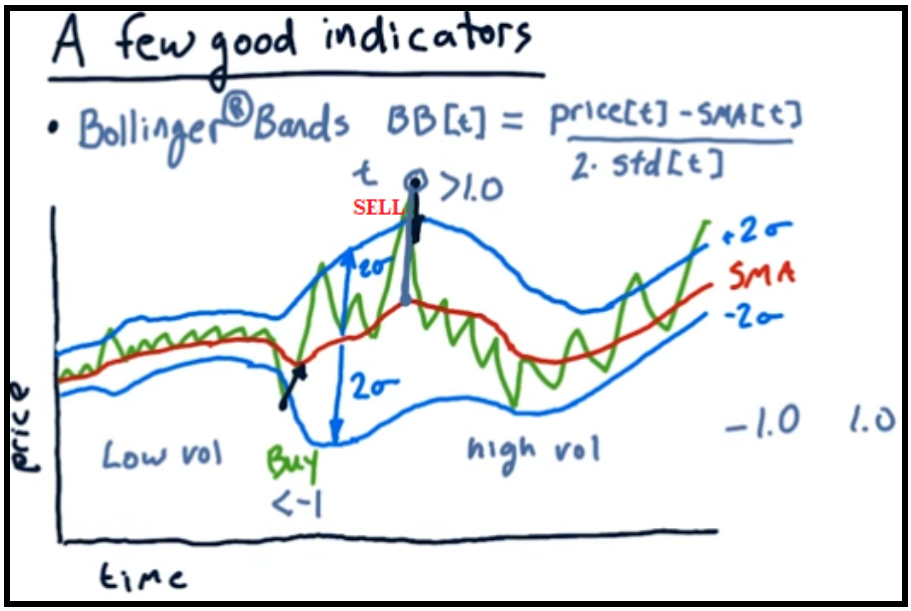
- First they look for places **where the current price crosses through the simple moving average**
  - Those tend to be important events especially if the average is over many many days
  - If you combine that with momentum, if the price has strong momentum and it's crossing through that simple moving average, that can be a trading signal
- The second way is as a **proxy for underlying value**
  - If you look back over a certain period of time and take that average price, that might represent the true value of the company
  - If we see a large excursion from that price, we should expect that the price will come down to that average - it's an arbitrage opportunity like we saw with the fundamental analysis
  - Buy and sell opportunities when there's a strong diversion from that moving average



- **The *neg* pts are buy opportunities and *pos* are sell opportunities!**

Bollinger™ Bands

- Regions of low and high volatility (vol is volatility not volume)
    - John Bollinger observed that for low volatility stocks or stocks that are currently experiencing low volatility - you probably want to use a smaller number for that trigger (the price trigger?) and for high volatility a larger number. That is accomplished using the standard deviation
- So we're taking the simple moving average, but let's add a band above and below that is two standard deviations (2-sigma) and that's our measure for how strong of a deviation we want to see before we respond to it



Buy or Sell?

- At 1, we go from outside to inside across the band so sell because it's the above band
- At 2 we go from inside to outside so nada
- At 3 we go from outside to inside again so buy because it's the below band
- At 4 is same as 3 we go from outside to inside in the below band so buy

Normalization

- Simple Moving Average (SMA) values from -.5 to +.5
- Momentum -.5 to +.5
- Bollinger Bands (BB) -1.0 to +1.0
    -
- These have different value ranges so they could overwhelm the other indicators and become the most important one
- It could be even worse if we included something like PE ratio (Price Earnings) that can range from 1 to 300
    -
- The solution is normalization
    - Normalization takes each of those factors and essentially compresses them or stretches them so that they vary on average from -1 to +1 and a mean of 0

normed = (values - mean)/values.std()

- Values are for a particular factor, subtract the mean from all of them and divide by the standard deviation of all of them

Technical Analysis wrap up

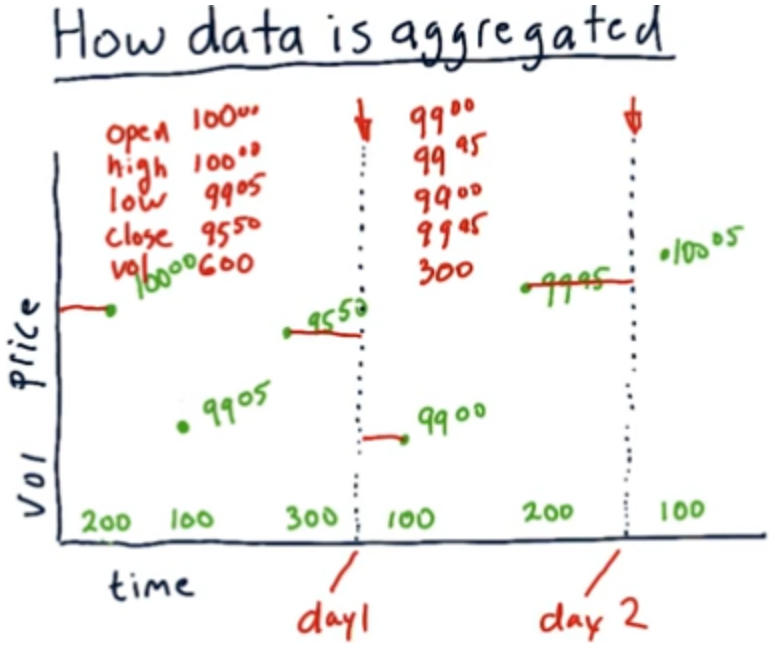
- **Heuristics**
    - **Technical indicators are really heuristics** that represent someone's interpretation or hunch on how a statistical approach to previous prices and volume might suggest future price movement
- **Tucker's approach**
    - The examples in this lesson
- **Don't start trading yet** no matter how excited you are about Technical Analysis

02-07 Dealing with data

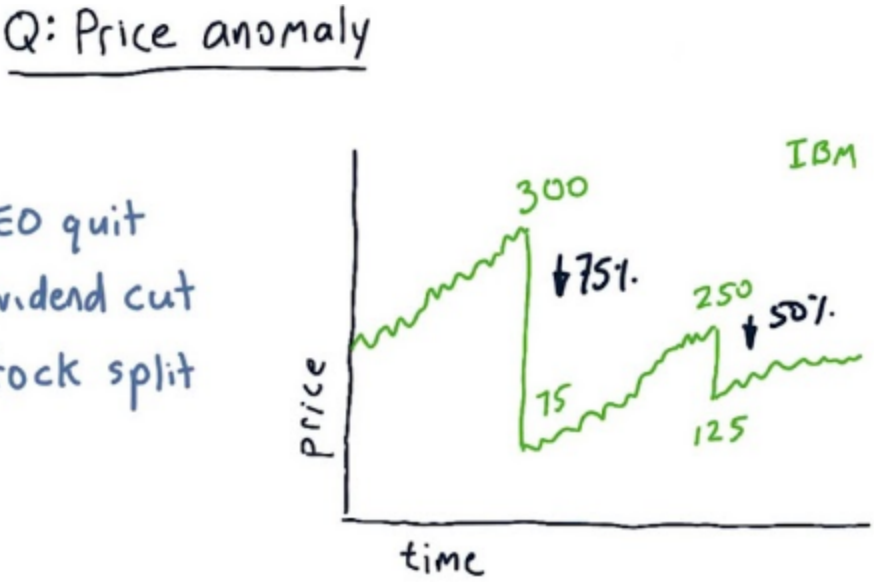
How is data aggregated?

The finest resolution of data is called a **tick**. A **tick** represents a successful buy sell match or a successful transaction

- Open is the first transaction within the time period
- High is the highest price
- Low is the lowest price
- Close is the last transaction
- Volume is the total volume during that time period



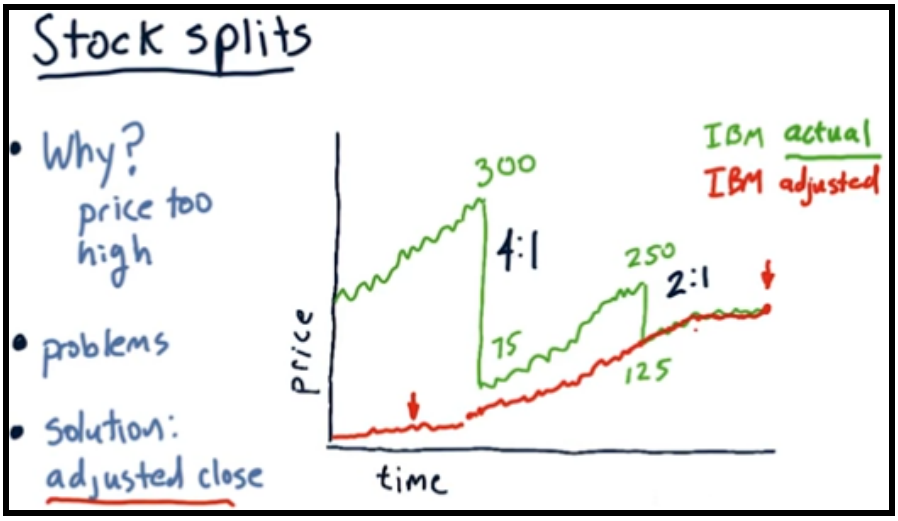
Price anomaly



- It's not CEO quit because: Although this is a possible cause for a sheer price drop, it is unlikely that the drop would be exactly 75% or 50%. Also note that the trend continues to be the same (here upward) immediately after the sheer drop. Would you expect this if the CEO quit?
- It's not dividend cut because: Yes, a dividend cut can result in a sharp price decline - simply because the stock is no longer as lucrative for investors. But you would expect to see a further downward trend, wouldn't you? Why would investors suddenly regain confidence in the stock immediately after the decline? Something else must be going on...
- Stock splits result in reduced price per unit. For instance, say each stock unit is divided into two units, then the value of each new unit drops to half. This is what likely happened.

Stock splits

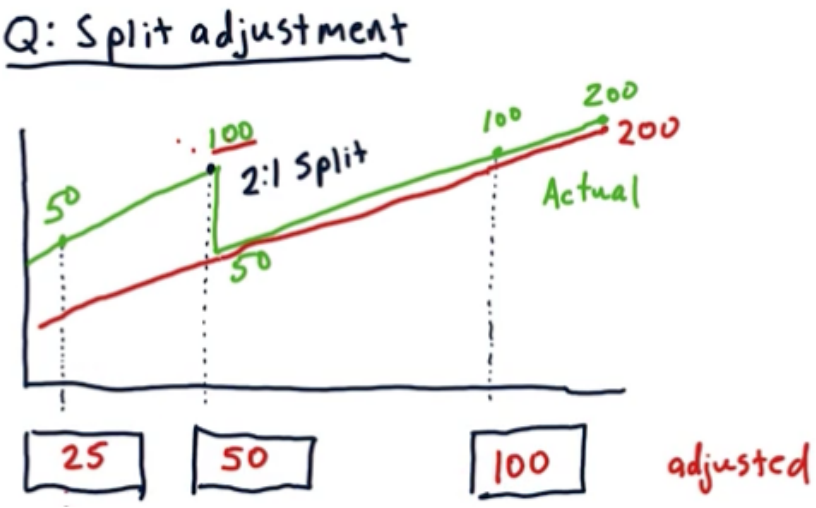
These happen because the price is too high

- **Why** is the stock price being too high a problem?
  - Because options are bought and sold in chunks of 100 and they become less liquid as they become really expensive
  - From the point of view of options and individual stock shares, very high prices are a problem
    - Even in the case when you want to buy just one share of the stock can be a problem (e.g. apple was $600)
  - If you're building a portfolio and you want to have a finely tuned proportion of each stock in your portfolio - if some of the stock prices are very high it becomes difficult to get that fine resolution that you want
  - So when the prices get very very high, what companies do is they say is look, let's take that one share that's priced at 300 and break it into 4 shares at 75 - this is called a 4 for 1 split for the 75% drop and 2 for 1 split for the 50% drop. (4:1 and 2:1)
- **Problems**
  - You can't use this actual data for shorting because the value of the stock doesn't actually decrease and you have to account for all the stock splits
- **Solution:**
  - **Adjusted close or adjusted prices**
  - 

At close actual and adjusted are always the same - then you go back in time and you start dividing by 2 when 2:1 and then by (2/4) when 4:1 in order to get a smooth accurate curve of the value increase.

Split Adjustment

“You would make a fine financial analyst :)” **awwwwwww**

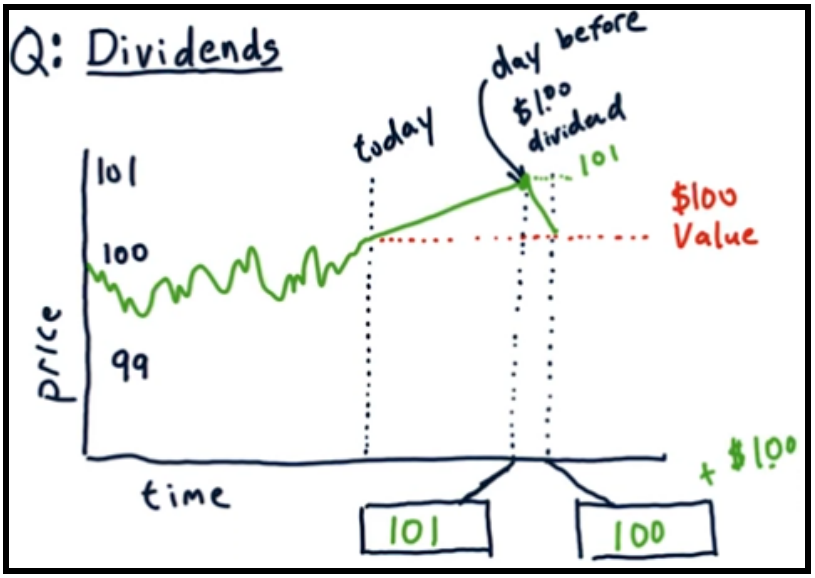


At the end the actual and adjusted are the same, then start dividing by 2 at 2:1. It doubled in value at the split and it quadrupled in value at close.

Dividends

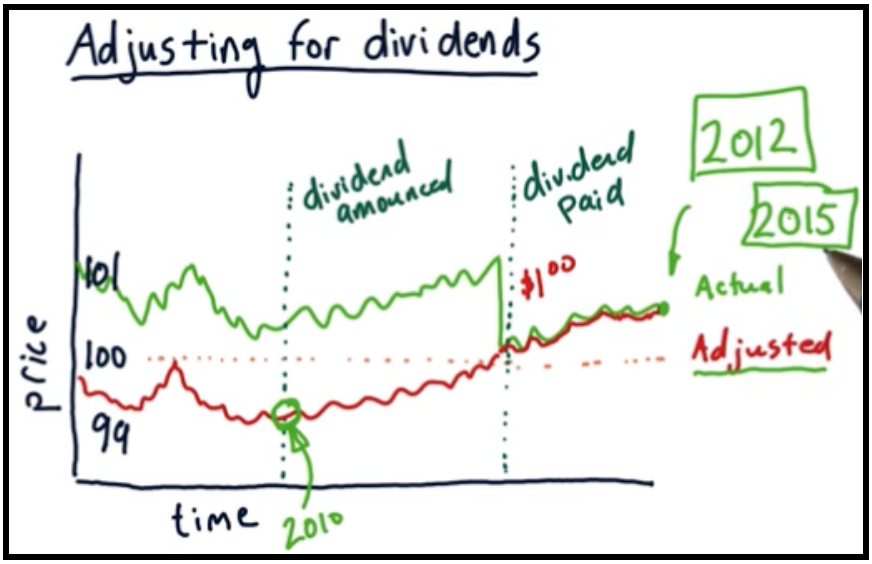
Compute company value based on dividends

- If you know that owning a particular stock on a certain day is going to give you $1 in dividends, would you like to own it?
- How does that affect the demand (and subsequently, price) of the stock?
- How about the day the dividend is paid out - would the demand for the stock remain the same after the payout?



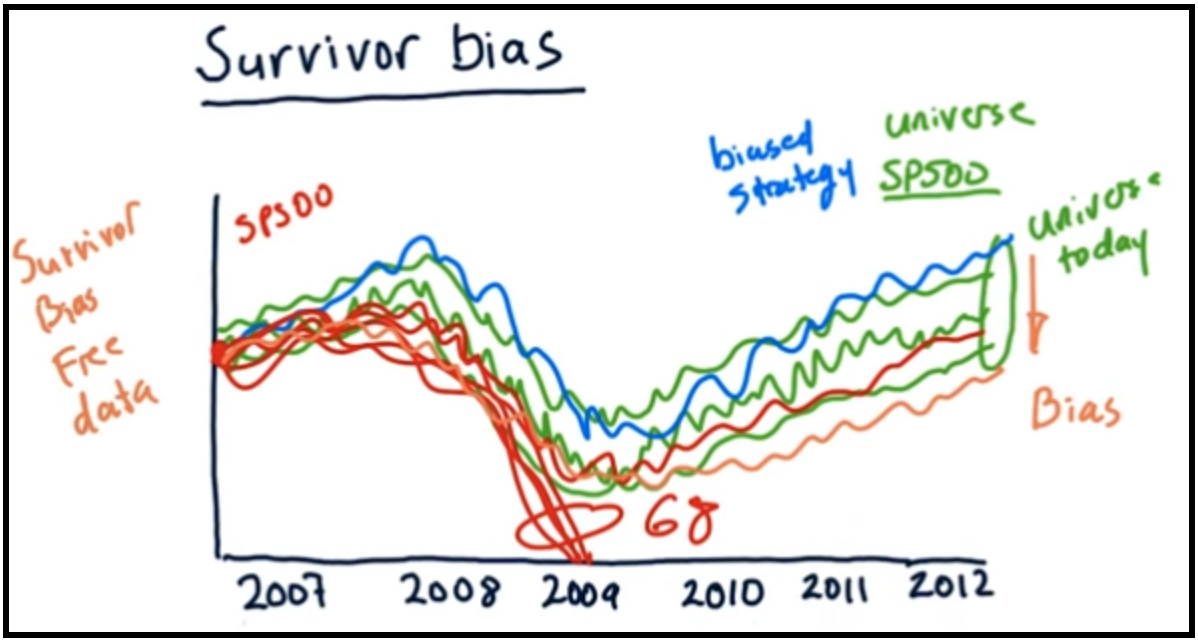
Adjusting for dividends

The same way as we adjust for splits. In this case we adjust the prices down by the proportion of the dividend, by about 1% if the stock is $100 and the dividend is $1.

- It's very important to always use adjusted close in calculations
    - Be aware that the adjusted close values depend on the current date when the data is queried/sampled because it takes into account all the dividends and splits that occurred in the meantime
- 

Survivor bias

- One of the things we do in this class is simulate strategies that we might develop, we roll back time and pretend that we traded on certain dates according to certain signals, and we see what the result of our strategy might have been.
    - To do that you have to start with some universal stocks and one of the most common universes is the SP500
    - When you simulate your trading:
        - You roll back time and you look at that universe of stocks
        - You apply your algorithm to choose which stocks you might buy
        - **A common mistake that people make is that they look at the membership of that universe as of today, then they go back in time and they use that list of stocks for their strategy**
            - But we know that those stocks survived in reality there are many that didn't
            - So you must use survivor bias free data which is not necessarily free of cost, but it's not very expensive either or so the Prof. says



- **68 stocks died from the universe in 2007 and the universe in 2012**

02-08 Efficient Markets Hypothesis (EMH)

So far we've been operating under some assumptions

- For TA Technical Analysis we assumed that there is information in historical price and volume data that we can discover and exploit in advance of the market
- For FA Fundamental Analysis we assume there is information in fundamental data like earning that can be exploited and traded upon in advance of the market

The Efficient Markets Hypothesis says we're wrong about both

Assumptions

- **Large number of investors operating in the market for profit**
  - So they have an incentive to find opportunities where the price of a stock is out of line with what it's true value is
  - Because there are so many of these investors operating simultaneously, any time a little bit of information comes out there, the price is going to move
- **New information arrives randomly**
  - It arrives at random times and at random rates for different stocks, but it's constantly arriving
- **Prices adjust quickly**
  - Investors are paying attention to that information and therefore the prices are adjusting quickly
- **Prices reflect all available information**
  - The current price reflects all available information
  - All the information that's trickling in is acted upon by the investors, the price adjust quickly to that information and the current price reflects all of the information about that stock

Where does information come from?

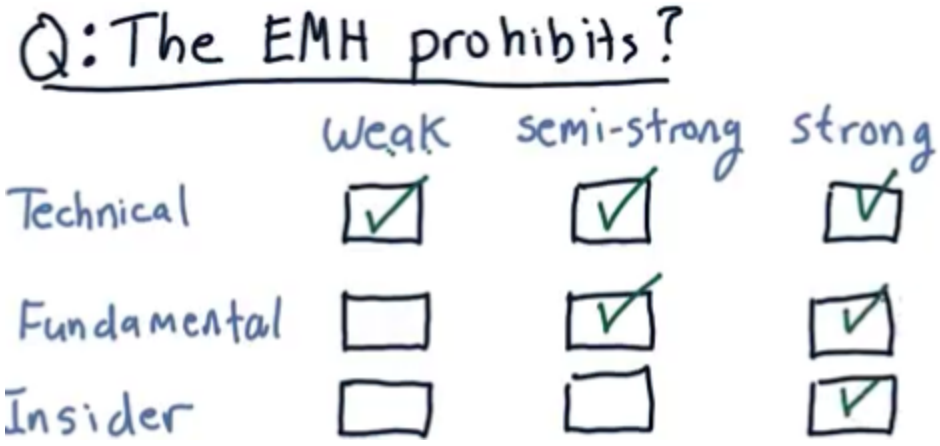
The following are sorted from most public to least public

- **Price/Volume**
  - It's public, it's rapid, it's quick everybody can see it
  - It's the basis for Technical Analysis
- **Fundamental**
  - Reported quarterly and everyone can see it as well because it's public but it points more to the root of the value of the company than just the price volume
- **Exogenous**
  - It's a fancy name for a simple concept - it's information about the world that affects the company (island data? Or ocean data? Or both?)
  - If we're looking at airline stock, an exogenous piece of data is the price of oil
    - If oil goes down, airline goes up because energy is the number one cost of airlines
- **Company Insiders**
  - A very important and very secretive type of information relates to company insiders
    - Suppose you're the CEO and you know that this drug that you've invented is about to be improved. You might go buy shares of your stock because you think the price of your stock is going to go up because that drug is going to be approved
    - Now depending on the circumstances, it may or may not be legal but this reflects information that you have that people outside the company do not have. This type is the least accessible of most other types of information as well

Three forms of the EMH (Efficient Market Hypothesis)

There are three versions of the official markets hypothesis that go from weak to strong and we'll take a look at them each one at a time

- **The weak form of the EMH**
    - Says that future prices cannot be predicted by analysis of previous (historical) prices
    - The idea here is that the current price reflects all the information we might know - so just by looking at these historical prices you can't predict what is going to happen next
    - This does leave room for **Fundamental Analysis**
    - It is silent on fundamental or insider information, this is all about price
        - You can't profit by looking at historical price
- **The semi-strong version of the EMH**
    - Suggests that prices adjust immediately to new public information
    - When companies have their quarterly reports that contain fundamental information, prices react immediately to that information
    - So if semi-strong is correct that would seem to prohibit even fundamental analysis because the prices adjust quickly ?!?
- **The Strong version of EMH**
    - Says we can't even make money on **insider** information - it can't be leveraged!
    - Prices reflect all information public and private - so even if there's some secret information within the company that points to a higher price later the price will go up in the face of that knowledge
    - If the strong version of the EMH is true, it is essentially impossible to make money by holding a portfolio other than the market portfolio

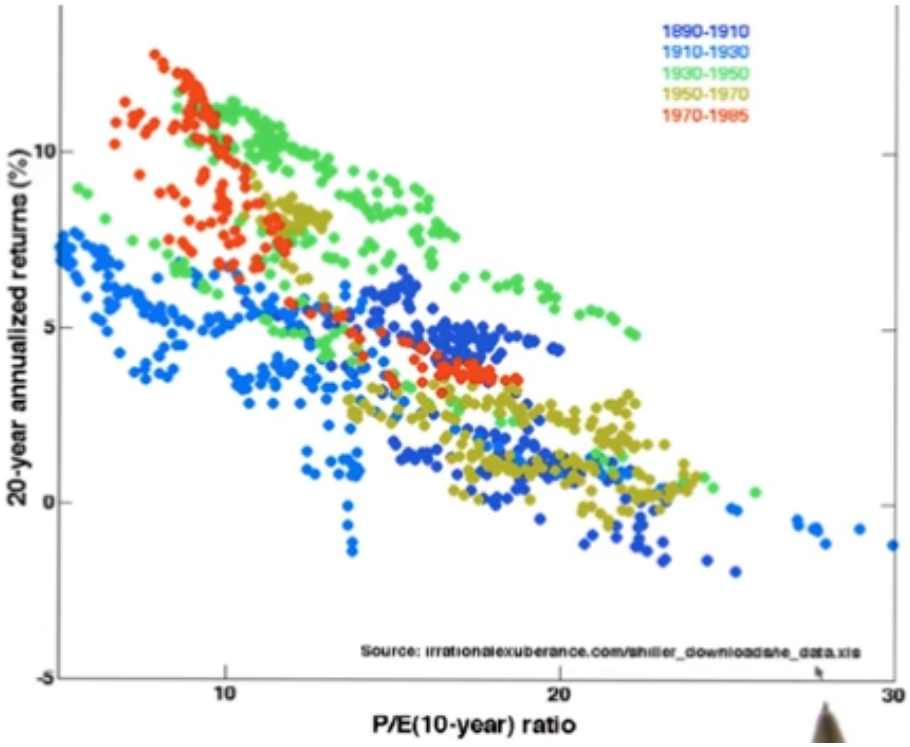


You take these ^^^ by column

- The weak version prohibits us from profiting from technical analysis because the weak version says that you can't predict future prices from past prices. But the weak version didn't say anything about fundamental or insider information
- Semi-strong prohibits technical analysis, also fundamental but it's silent on insider
- The strong version says you can't profit from any of these three types of analysis

Is EMH correct?

- If it is, then a lot of the things we're trying to do in this course will not be possible
    - Namely we can't beat the market using any of the strategies we might be looking at
- There are a few versions of EMH that are not correct and there are a number of successful hedge funds out there that would seem to indicate that you can make money in the market by investing in things other than just the market portfolio
    - The strong version of the efficient markets hypothesis is the least solid
        - In other words the strong version says that you can't even profit from insider information
            - The reason that this ^^^ above one is not true is because we have seen people make money from insider information (some have gone to jail) but clearly it's a method that can provide profit even though it might be illegal
- The following is a dataset that seems to refute the semi-strong version of the efficient markets hypothesis as well



- **P/E price/earnings**
- Let's look at the blue dots. Each dot here represents the P/E ratio on a particular date, so for instance this stock had a P/E ratio of 20 that's price to earnings and is the horizontal component of its location.
- The vertical location is how much money did it make in terms of price over 20 years
- The blue stock dot at 20,5 had a price/earning ratio of 20 and it made about 4% annualized returns
- As we go down this way the P/E ratio is lower and lower numbers for P/E are better
    - If you think about it, price divided by earnings, if it's lower price it has higher earnings you get a lower number so lower values indicate higher value for a stock and each group of colors represents a different decade when the analysis was done so for instance this dark blue (1890-1910)
    - For all of these decades we saw that at the beginning of the corresponding time period low P/E ratios corresponded with higher returns so this shows that price/earnings ratio are very predictive across many many decades of future returns and that tends to refute the semi-strong version of the EMH

02-09 The fundamental law of active portfolio management

"Only when the tide goes out do you discover who's been swimming naked" - Warren Buffett

Wide diversification is only necessary when investors do not know what they're doing

- Investor skill
- Breadth or the number of investments

Grinold's Fundamental Law

Grinold was seeking a method of relating performance, skill, and breadth in investing

- Performance
- Skill
- Breadth

You may have the skill, but not a lot of opportunities or breadth to exercise that skill.

performance = skill * sqrt(breadth)

IR = IC * sqrt(BR)

- Performance can be summarized into something called **Information Ratio IR** which is very much like the Sharpe Ratio that we discussed before but it refers to the sharpe ratio of excess returns - in other words the the manner in which the portfolio manager is exceeding the market's performance.
- **Skill** is summarized in something called **Information Coefficient IC**
- **Breadth BR** is just how many trading opportunities we have

The coin flipping casino

- Flip coins instead of stocks
- The coin is biased-like **Alpha** *0.51* heads more likely to come up heads than tails
- Uncertainty is like **Beta**

Betting

- **Bet N coins**
    - Win: now have 2*N
    - Lose: now have 0

Casino

- **1000 tables**
- **1000 tokens**
- **Games runs in parallel**

Coin Flip Casino: Reward

- Expected Return
- **Single bet:** chance that we'll win the bet times the return we get plus the chance that we would lose the bet times what we would lose the best times what we would lose
    - **0.51 * 1000 + 0.49 * (-1000) = $20**
- **Multi bet:** the chance is now of winning or losing a dollar instead of 1000
    - **1000 * (0.51 * 1 + 0.49 * (-1)) = $20**

Even if these yield the same expected return, the muti bet approach is better because of the risk

Coin Flip Casino: Risk 1

- The risk to Lose it all
    - Single bet 49%
    - Multi bet .49^1000 = **1.57E-310%**
        - **It's so small is not even a number (nan)**

Coin Flip Casino: Risk 2

- Consider the standard deviation of all those individual bets
- Look at the std of
    - One token per table
        - -1,1,-1,1,1 … -1 across all tables - **1** is gain **-1** is loss
            - stdev = 1.0
    - One 1000 token bet, 999 0 token bet, win or lose:
        - stdev(1000,0,0,0,0,0,0) = 31.62
        - stdev(-1000,0,0,0,0,0,0) = 31.62

Coin Flip Casino: Reward/Risk

- Risk adjusted reward just like Sharpe ratio
- Single bet case = $0.63
- Multi bet case = $20

Coin Flip Casino Observation

We have two extremes

- We bet all 1000 chips on a single coin flip
    - **Sharpe ratio = 0.63 = SRsingle (the risk adjusted reward)**
- We bet 1 chip on each of 1000 coin flips
    - **SRmulti = $20**

BR = breadth, opportunities, coin flips!

SRmulti = SRsingle * SQRT(BR)

$$SR_{multi} = SR_{single}\sqrt{bets}$$

performance = skill * sqrt(breadth)

- It turns out that in general if you carry that scenario out to more examples, that if you split your bets evenly across multiple tables, this relationship will hold - in other words the Sharpe Ratio for the single bet 1000 chips on one table is sort of our base case and as we spread it out over more and more tables, the Sharpe Ratio improves by the square root of that number of bets

Coin Flip Casino Lessons

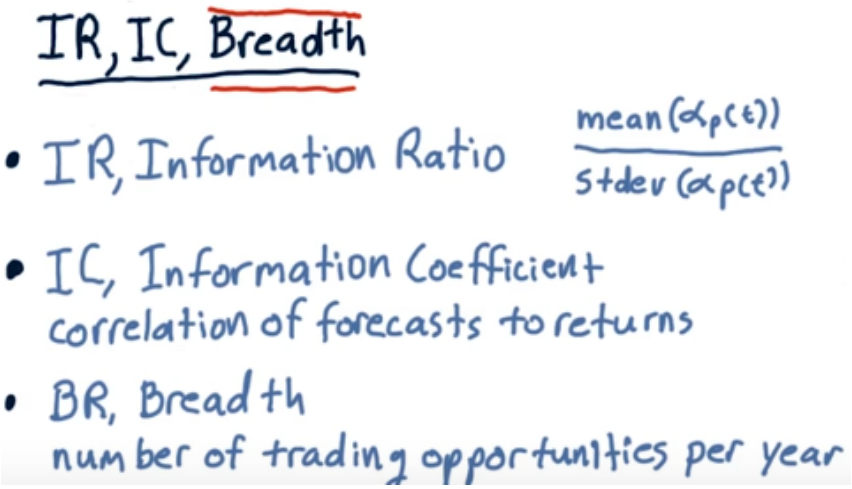
When we consider the risk and reward together, we come up with three lessons:

- Higher **Alpha** generates a higher **Sharpe Ratio**
- More execution opportunities provide a higher **Sharpe Ratio**
- **Sharpe Ratio** grows as the square root of breadth

Real World, in particular Hedge Funds

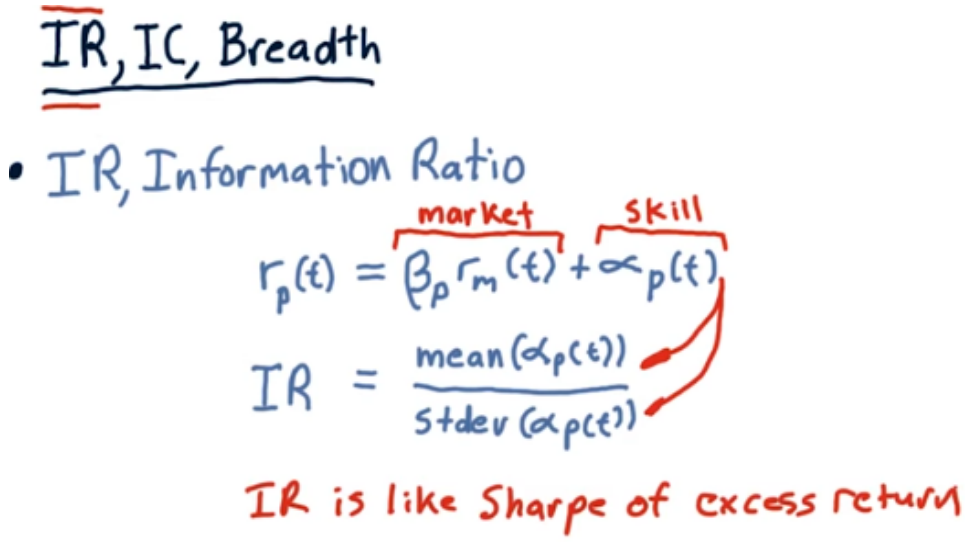
- These have similar performance
    - RenTec trades 100k/day
    - Warren Buffett holds 120 stocks and rarely trades
- Can a single theory relate the two?
    - **Yes and it is the fundamental law of active portfolio management**

IR,IC,BR



- **IR Information Ratio**
  - People use Information Ratio as a measure of manager's performance all the time
  - It's the Sharpe Ratio of excess return
- **IC Information Coefficient**
  - Is the correlation of the manager's forecast to actual returns, range from 0.0 (no correlation) to 1.0 (strong)
- **BR Breadth**
  - Represents the number of trading opportunities per year
    - 120 for Warren Buffett
    - 100k * 252 days for Jim Simons
- **All these are oriented around an annual measure**

The return on our portfolio on a particular day is equal to the market component of the return which is **Beta** for that portfolio times the return on the market for that day, plus this residual return **Alpha**



The Fundamental Law IR = IC*SQRT(BR)

$$IR = IC \cdot \sqrt{BR}$$

IR: perf; IC: skill; BR: breadth

Q: Simons vs Buffet

- Both have same IR
- Simons' algo is 1/1000 as smart as Buffet
- Buffet trades 120/year

How many trades must Simons execute?

$$IC_B \cdot \sqrt{120} = IC_S \sqrt{X}$$

$$IC_B \cdot \sqrt{120} = IC_{B/1000} \cdot \sqrt{X}$$

$$1000 \cdot \sqrt{120} = \sqrt{X}$$

$$1000^2 \cdot 120 = X$$

$$120,000,000$$

120 000 000

02-10 Portfolio Optimization and the Efficient Frontier

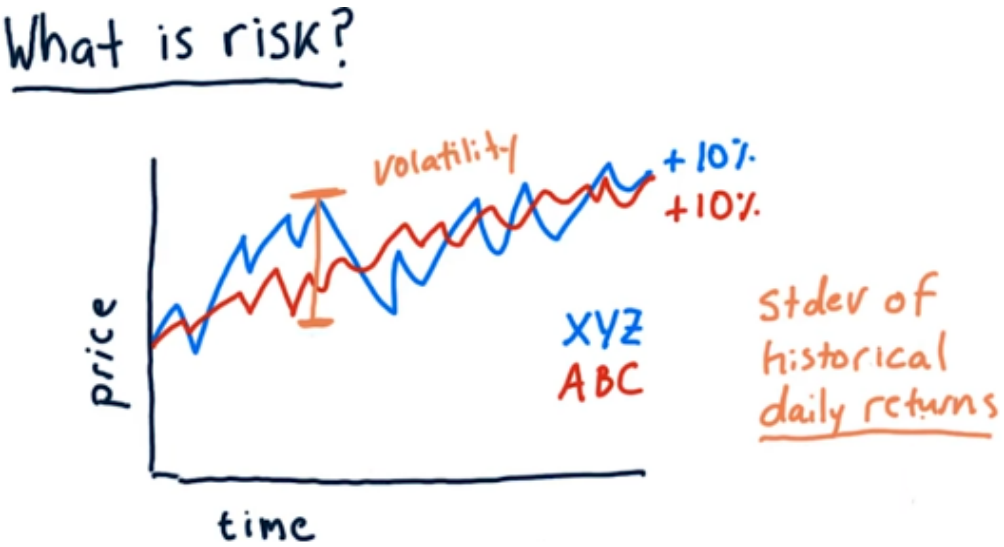
Suppose you have a set of stocks that you've determined are good investments. How much of your portfolio should you invest in each?

In this lesson we take a look at an approach called mean variance optimization, or portfolio optimization.

The specific question we're looking to answer is this:

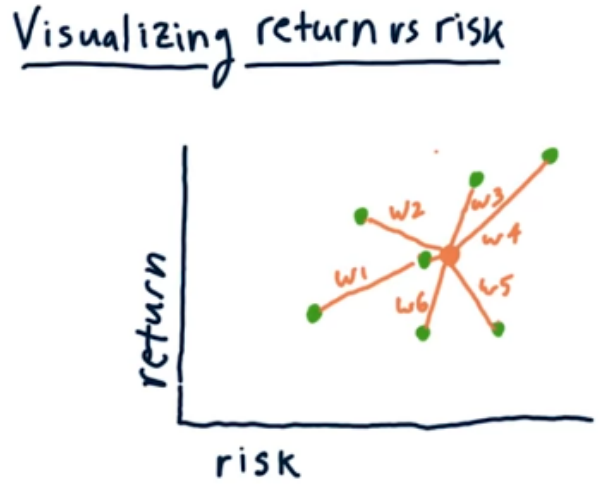
- Given:
  - Set of equities
  - Target return
- Find:
  - Allocation to each equity that minimizes risk

What is risk?

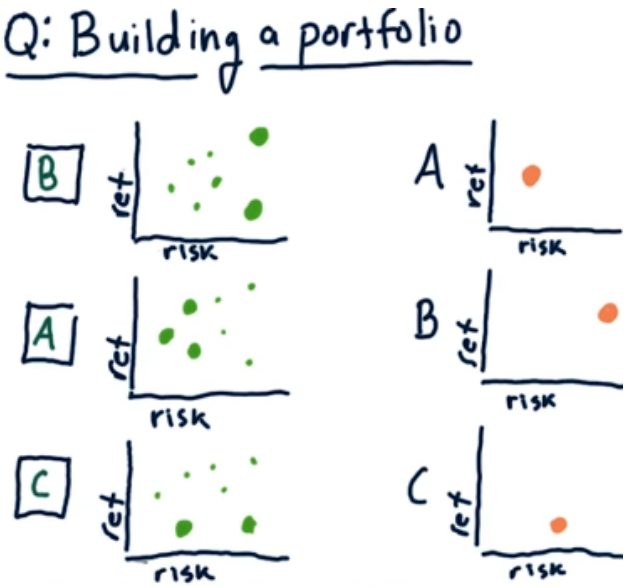


- **It is the volatility that we use as a measure of Risk**
  - **This is simply the standard deviation of historical daily returns**
  - **It's the standard view of risk in most finance texts**
- There are others that we might touch on here and there but the key is the standard deviation of historical daily returns

Visualizing Return vs. Risk



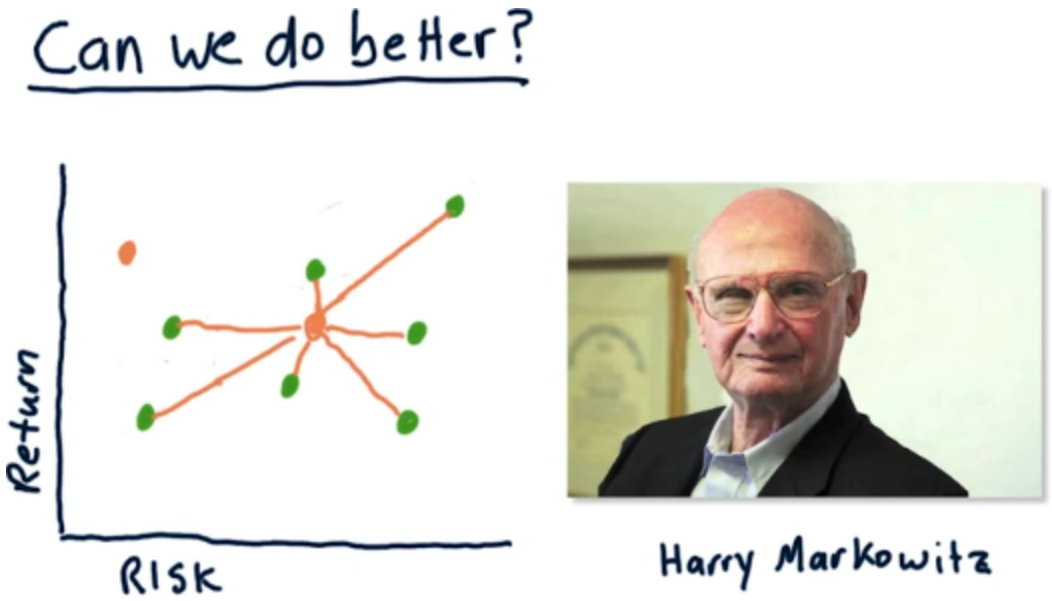
- **w_i are weights, allocation**
- **green dots are different stocks**
- **The full graphic is the portfolio with weight allocations**
- **The center orange dot is the total risk and return for the portfolio**



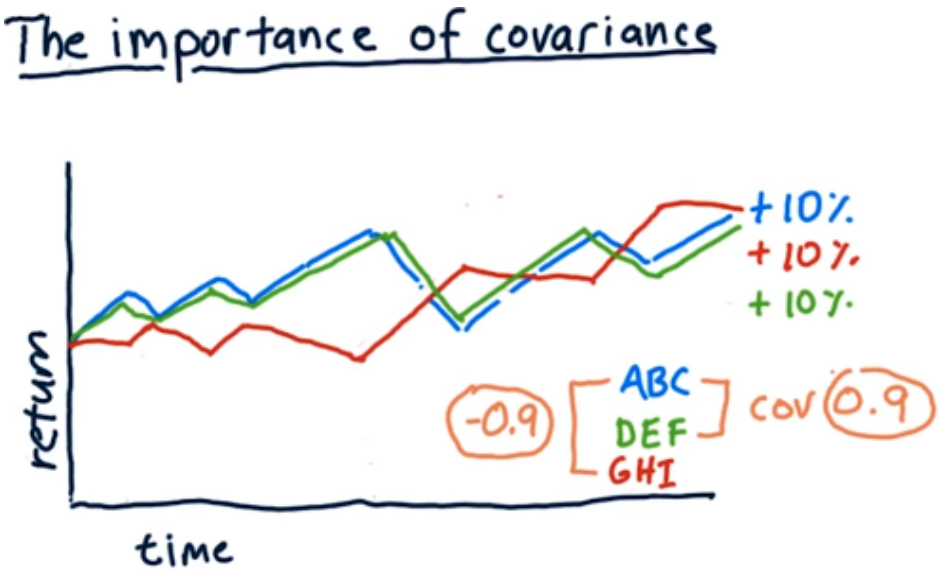
- **the weights are the size of the green dots**
- **the risk return level is the size of the orange dots**

Can we do better?

- **Yes! Because of Harry Markowitz (Nobel Prize)**
    - What he discovered and what people have been overlooking was the relationship between stocks in terms of **covariance (correlation)** - so the resulting performance of a portfolio, especially in terms of risk is not just a factor or a blend of the various risks, but it has to do with how they interact day to day
    - So indeed if we pick the right stocks in the right proportions we can get a portfolio that performs over here (in the upper left) that can have lower risk than any of the individual assets

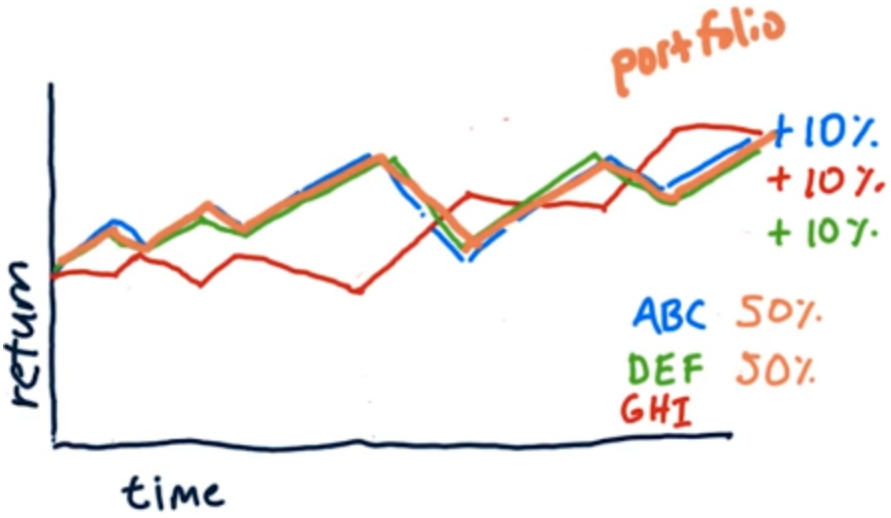


- Up until his discovery, most people viewed bonds as the lowest risk asset, if you wanted low risk, you should use bonds only
- Markowitz showed that a blend of stocks and bonds is actually lower risk than either of those (stocks or bonds) by themselves



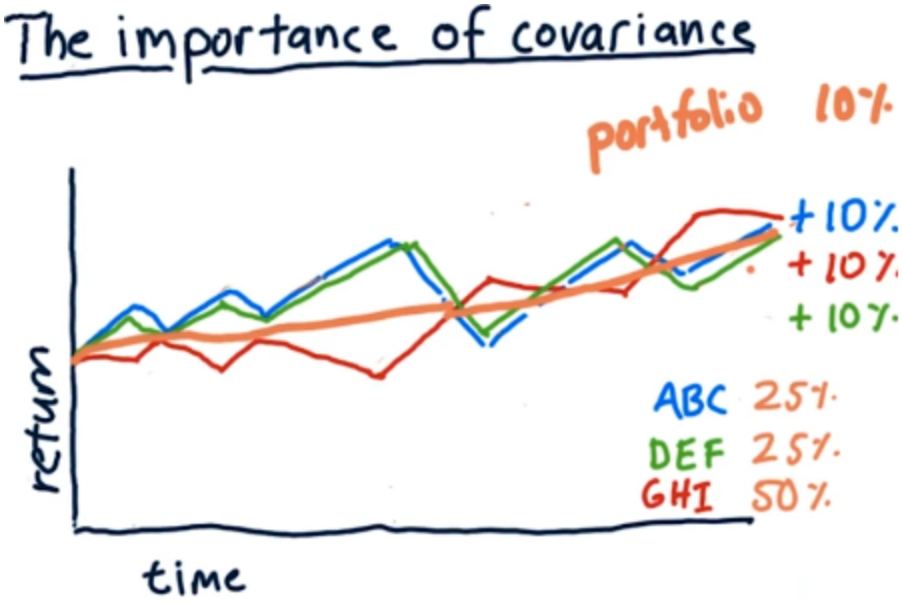
- **ABC and DEF have positive covariance, correlation (both go in the same direction)**
- **ABC and GHI have negative covariance, correlation (when one goes up, the other goes down)**

Best way to blend together



- **With 50% allocated to ABC and 50% allocated to DEF**
- **Our portfolio (thick orange line) returns 10%**

Another way



- **With 25% allocated to ABC and 25% allocated to DEF and 50% to GHI we have a nice smooth blend of the three that has very low volatility and this is the magic that Markowitz provided**

Mean Variance Optimization (MVO)

- What Markowitz added to the game was this consideration of variance and covariance between individual stocks and the recognition that you want to blend those together that have anti-correlation so you can have a much lower risk portfolio if you combine assets that are anti-correlated or anti-varianced (if that's a word)
    - Because when one moves up, the other moves down, they cancel each other out and you have much less volatility
- In general overall you want these assets to move up together
    - So often what we are looking for is anti-correlation in the short term and positive correlation in a longer term
- Out of this work grew a number of algorithms and one of the key ones being **Mean Variance Optimization MVO**
    - Which is a way of taking a potential set of assets and figuring out how they should be blended together by looking at their covariance among other things

Let's look at these assets and how we might combine them, how to allocate funds to them to provide a good portfolio

- ***Generally the higher return stocks or assets whatever they might be, tend to also be the highest risk - so as we roll down the risk we tend to also look at lower return***

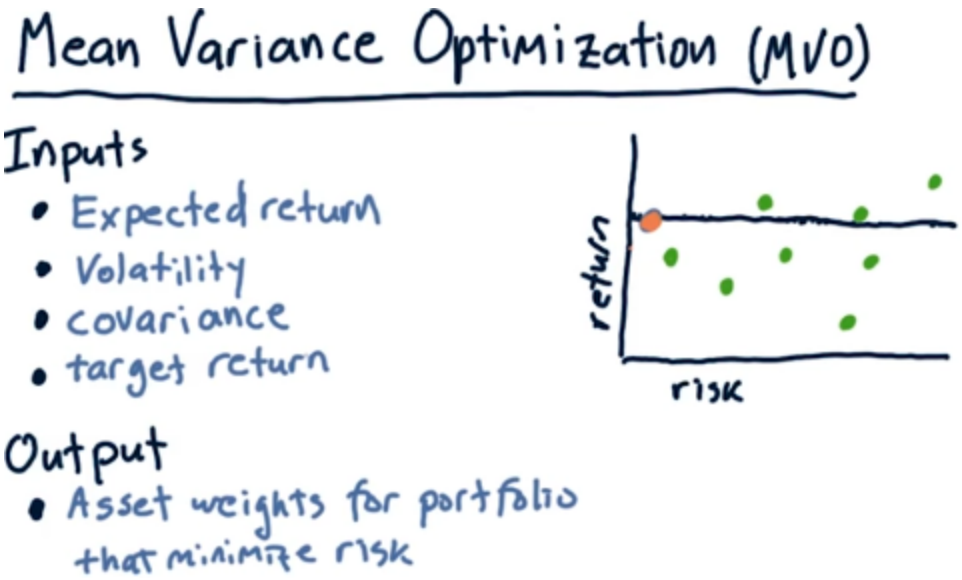
To figure out how they should be blended together:

Inputs

- **Expected Return**
    - For each stock - in other words what do we think the future it's going to provide in terms of return
- **Volatility**
    - Is simply historically how volatile has each one of these assets been
- **Covariance**
    - Is a matrix which shows between each asset and every other asset what is the correlation of daily returns
- **Target Return**
    - We can target a return anywhere from the max return asset to the min return asset and anything else in between those of course we can accomplish by blending

Outputs

- **A set of weights**
    - One weight for each asset that provides the target return but minimizes risk



The Efficient Frontier

- For any particular return level there is an optimum portfolio
- The orange dot reflects weightings of all the assets (green dots) that provides the lowest risk for this particular return (the orange dot again)
- What the efficient frontier means is that there are no portfolios above it or to the left (orange arrows up or left) and any portfolio below it is suboptimal in some way
- **If you draw a tangent line from the origin to the frontier, where the tangent line hits the efficient frontier, is where the max Sharpe Ratio portfolio for all of these assets**

In practice the efficient frontier isn't used for that much other than as a theoretical device, but people do often like to plot the efficient frontier so they can see where their portfolio is in relation to the assets that they're using and where they could be in terms of efficiency.

03-01 How Machine Learning is Used at a Hedge Fund

Introduce how hedge funds and other financial institutions utilize machine learning.

- In general the focus is on creating a model that can be used to predict future prices for stocks or other assets
- Models like these have been around for a long time, what's different about machine learning is that it provides a suite of tools that support a data-centric way to build predictive models

The ML Problem

- Scientists like to talk about the algorithms that they build in terms of the problems that they solve so this mini course is about Machine Learning and let's think about the problem the machine learning solves

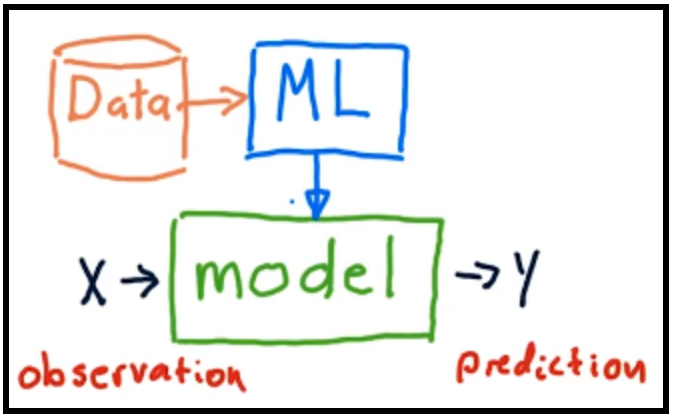
- **In most cases machine learning algorithms are focused on building a model.**

What's a Model?

- **A model is something that takes in observations X and run it through some sort of process and provide a y.**
    - y is typically a prediction and and X is some sort of observation of the world.
        - e.g. X are some features of stocks and y is a future price
    - **X can be multidimensional**
        - In other words there might be multiple factors that we're considering
            - Might be Bolinger Bands, PE (Price Earnings) Ratio and so on
    - **y is typically single dimension**
        - Just represents that single-dimension prediction that we're trying to make

There are other models that don't use machine learning at all

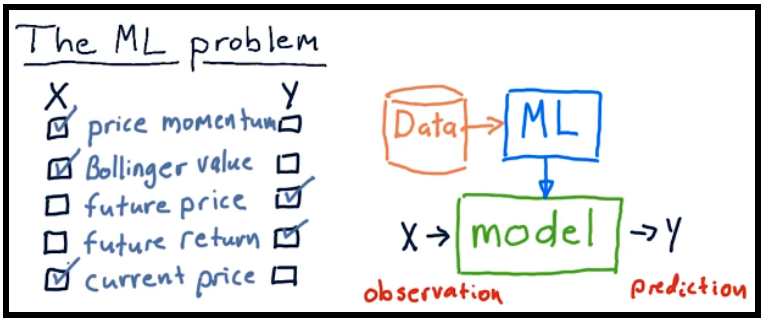
- For example the Black-Scholes Model that predicts option prices
- Others use math formulas
- With machine learning we're trying to use data



- **The machine learning process is to take historical data, run it through a machine learning algorithms of some sort to generate the model**
- **Then at runtime when we need to use the model, we push X's in it and y's come out**

Quiz

Consider you were building a model and we were going to use it in trading in some way. Which factors might you consider as inputs or X's versus which of these might be appropriate outputs of y's for our model?

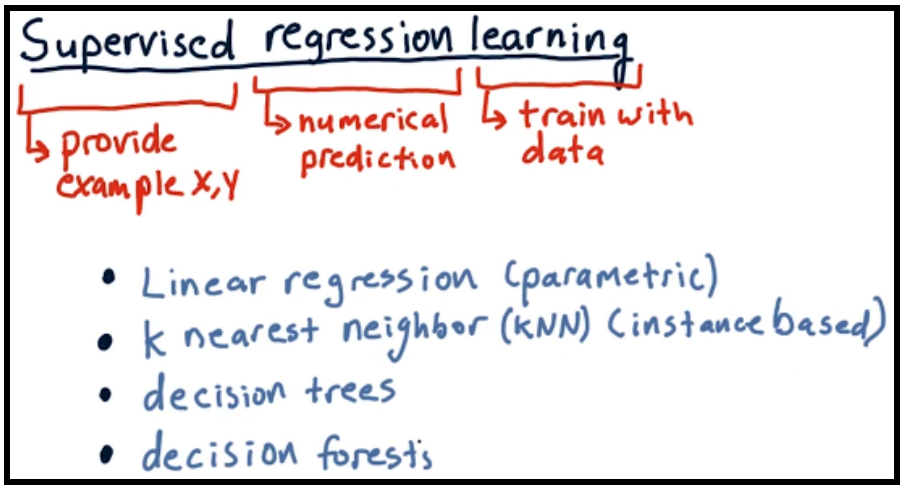


Supervised Regression Learning (SRL)

- **Supervised**
  - Means that we show the machine X and the and, if you will, the correct answer y
  - In fact we show the machine many, many examples of X and y and that's how it learns
    - This is the supervised component
    - OK, when I see this X this is the y that's associated with it
- **Regression**
  - Means that we're trying to make a Numerical Approximation or Numerical Prediction
  - That's as opposed to classification learning where we might be trying to classify an object into one of several types as opposed to making a numerical prediction
- **Learning**
  - What we mean by learning is that we are training with data
    - In this class we're taking historical stock data and training the system to make a prediction about the future, usually about price

SRL Algorithms

- **Linear Regression**
  - parametric - we take the data, munge it around to come up with a few parameters and then throw the data away
- **K nearest neighbor (KNN)**
  - Instance-based - we keep all this historic data, the X and y pairs and when it's time to make a prediction, we consult that data and this is what makes it instance based
- **Decision Trees**
  - They store a tree structure and when a query comes in it essentially bounces down that tree according to factors of the data
  - Each node in the tree represents essentially a question: Is this X value greater than or less than this other value?
  - Eventually we reach a leaf and that is the regression value that's returned
- **Decision Forest**
  - Are simply lots and lots of decision trees taken together and you query each one to get an overall result

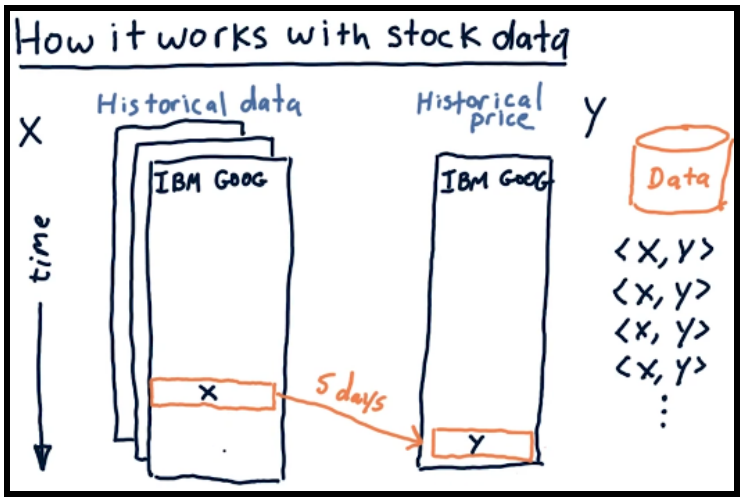


Robot Navigation Example

LAGR - learning applied to ground robotics

- Using KNN to learn how to navigate
- **Inputs**
  - In which directions are there obstacles
  - Direction towards the goal
  - X are these perceptions, what do I see around me and what's the direction to the goal?
- **Output**(s)
  - y or the output is which direction to steer
- The model maps these perceptions X to y, what it should do
  - In finance instead of y being which way the robot should go, it's often a prediction about what a future price is

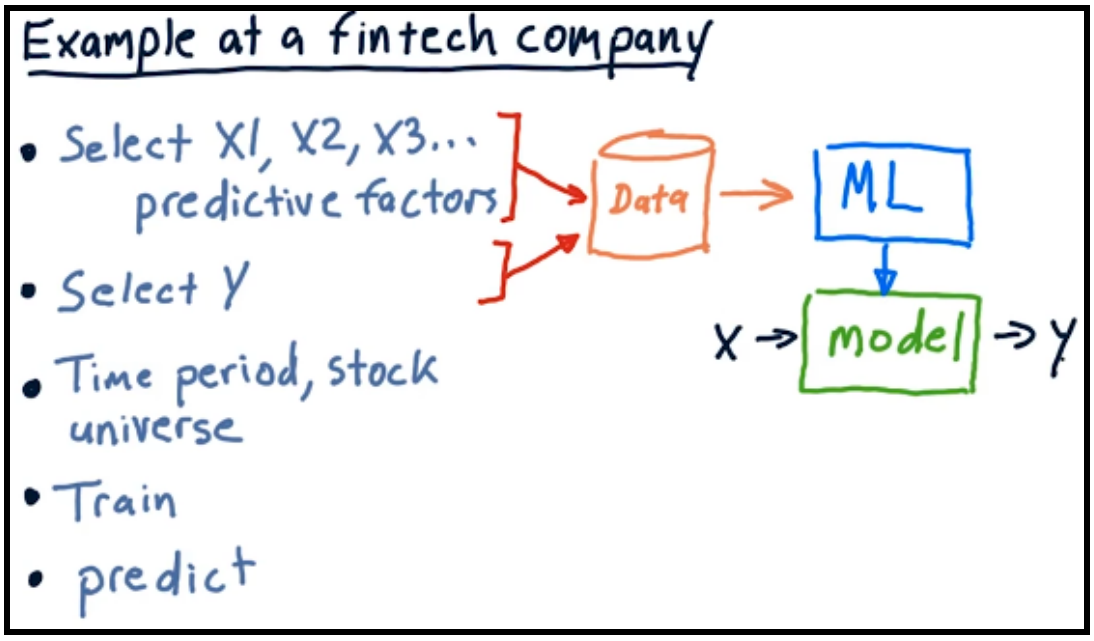
How it works with stock data



These are pandas data frames that contain some factors or features of stocks and it's arranged the usual way where each column represents the value of the feature for a particular stock. And time goes downwards. Each frame is for a feature. There is a dt of 5 days that we're trying to train and predict on.

Example at a FinTech company - Lucena Research

1. Select Xs, which factors we want to use, these are our Xs
    a. Predictive factors, X1, X2, X3
        i. Measurable quantities about a company that can be predictive of its stock price
        ii. Bollinger bands, PE ratio and so on
2. Select y, what is it you want to predict?
    a. Usually we want to predict
        i. Change in price
        ii. Market relative change in price
        iii. Or just a future price
3. X, y become our data that we use to train the model
4. Consider the breadth and depth of the data that we're going to use to train the system with
    a. Includes the time period, how far back in time do you want to go train the system
    b. What's your stock universe
    c. What universe of data
    d. Which symbols are you going to use to train the system as well
5. Train our model
    a. We unleash our machine learning algorithm that takes the data and converts it into a model
        i. KNN
        ii. Linear Regression
        iii. Decision Tree
6. Predict
    a. We take the model and do some prediction and the way we do that is we measure the quantities about the stocks that we want to make a prediction for now
    b. We measure what those Xs are today
    c. Plug those into the model and the model should provide us our y or our prediction



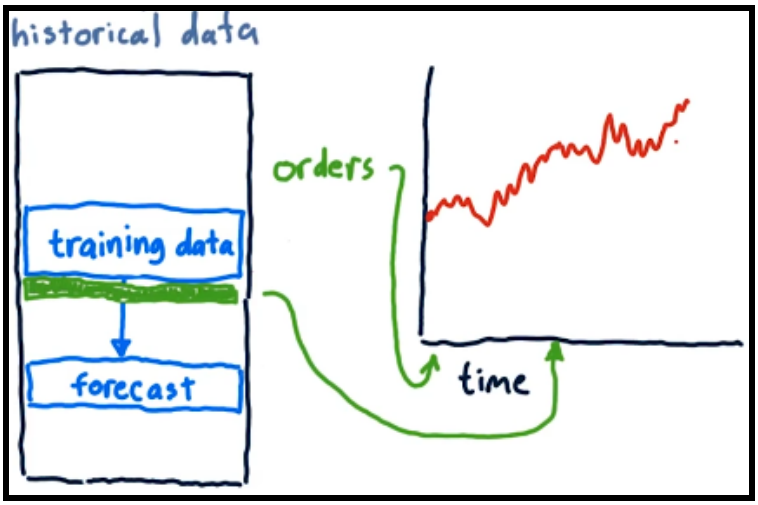
Price forecasting (demo)

https://classroom.udacity.com/courses/ud501/lessons/4684695874/concepts/46403887880923

Backtesting

The natural question is, how accurate are these forecasts? Can you act on them? Do they really predict the future?

- Well at least a first step towards answering that question can be found by back testing
    - You roll back the time and test your system
    - Historical data organized as usual with time coming down
    - To test the capability we need to limit the data it sees to a certain amount of time and then make predictions into the simulated future
- So we allow our system to only look at data up to a certain point
    - It can use data before that all it likes
- It builds a model
- Then makes a forecasts
- On the basis of that forecast we can then place orders anticipating that forecast will be achieved so we might long some stocks or short them as appropriate
- We can take those orders now and put them into our trading simulator and see how the portfolio works



ML Tool in use (demo)

https://classroom.udacity.com/courses/ud501/lessons/4684695874/concepts/46403887900923

Problems with regression

As you saw by the back-test, regression based forecasting can be useful
It's also worth noting that that particular back-test was not spectacular because it beat the S&P500
Usually we find that performance in the real world is not as awesome as in back testing
So we will probably see good return from that strategy but it wouldn't be quite the same as we saw in that back test

- **Noisy and uncertain**
    - There is value in there but it has to be accumulated over many trading opportunities
    - It's hard to know how confident you should be in a forecast, we could look at the standard deviation of the nearest neighbors which works OK but it's not really too strong of a measure however
- **Challenging to estimate with confidence**
    - So it's difficult to know how confident you ought to be in any particular forecast
        - It would be nice if you could now because that would enable you to essentially bet less on forecasts that are less certain
- **Holding time, allocation**
    - Additionally it's not clear how long you should hold a position that might have arisen from a forecast and how you should allocate to that position

Some of the above issues can be addressed using Reinforcement Learning where instead of making a forecast of a future price, we had the system learn a policy and the policy tells the system whether to buy or sell stock.

Problem we will focus on

- We're going to look at a certain period of data
- Train our models over that period
- Make forecasts and trade over some other period

- We will use the period of 2009 as our data to train our model
- You'll be implementing several machine learning algorithms to create different models and will be comparing them one against another
- Then we'll test over the years 2010 and 2011 so those testing values will become our Xs which will push through the model to create y a forecast
- Using this forecast you'll generate an orders.txt file which you can push through your market simulator
- We'll see how that strategy performs, measure it's Sharpe Ratio and it's total return and so on and that way we'll be able to compare different machine learning algorithms that generate these orders

03-02 Regression

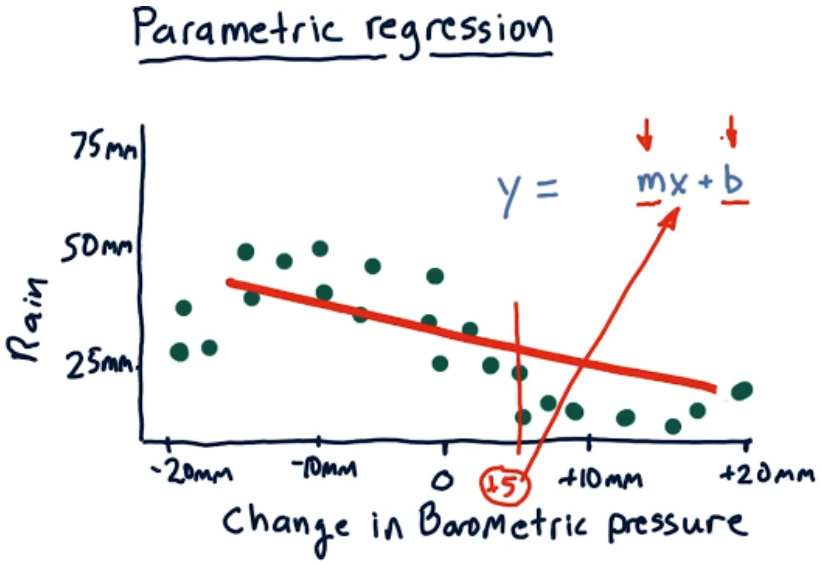
This lesson is about supervised regression learning (we prefer numerical model). In any case we're stuck with regression as the name for using data to build a model that predicts a numerical output based on a set of numerical inputs

Parametric Regression

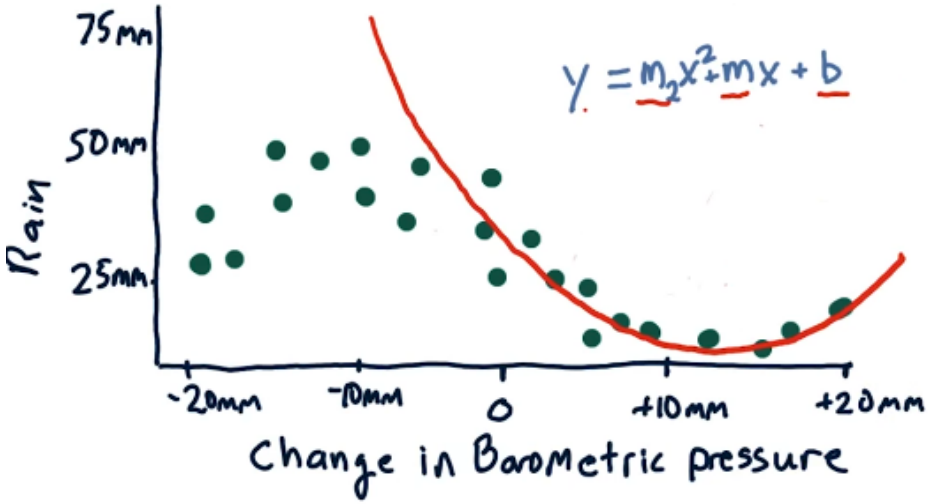
- This is a way of building a model where we represent the model with a number of parameters
- Suppose we want to build a model that will predict how much it will rain today based on changes in barometric pressure

- If barometric pressure declines that usually means there's bad weather coming and it's going to rain
- When it increases it means that we have good weather coming
- On the scatter plot each individual point represents one day

The classic solution to this problem is to fit a line to the data and this approach is called Linear Regression



- But it doesn't quite represent the data so we add another term, an x^2

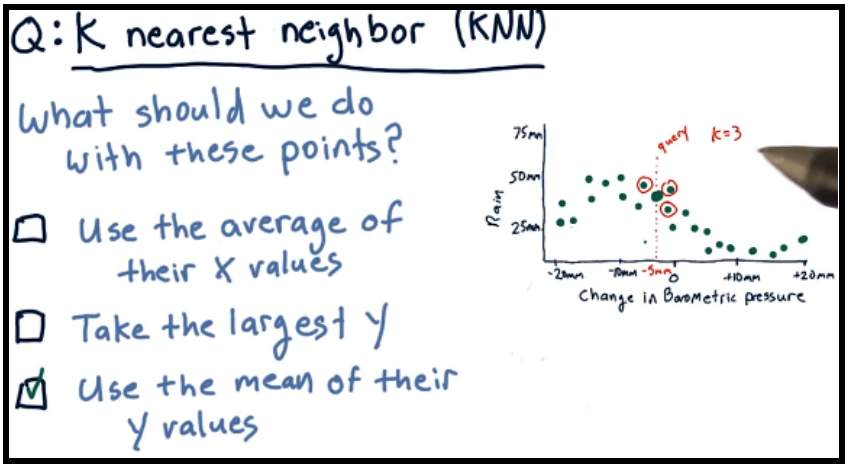


- but it's still not good so we keep adding m and x terms.

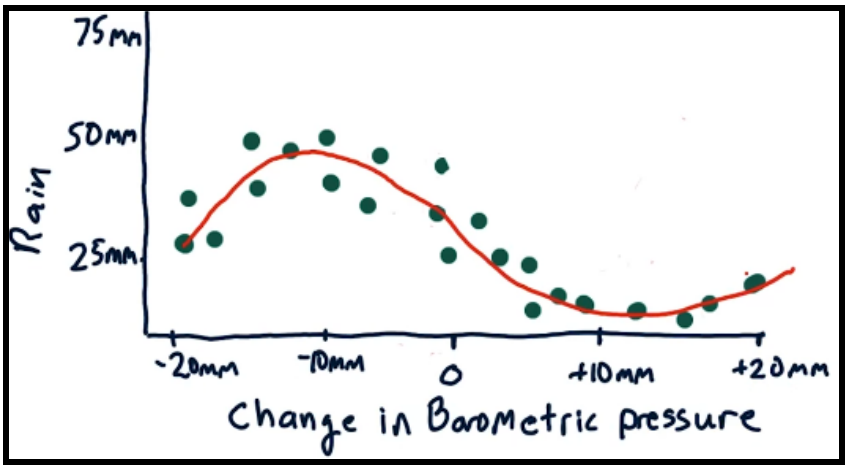
There's another way to approach this and that is

K nearest neighbor (KNN)

Which is a data centered approach, or instance based approach where we keep the data and we use it when we make a query.



- **If we keep doing this for a bunch of points we fit a line very nicely:**

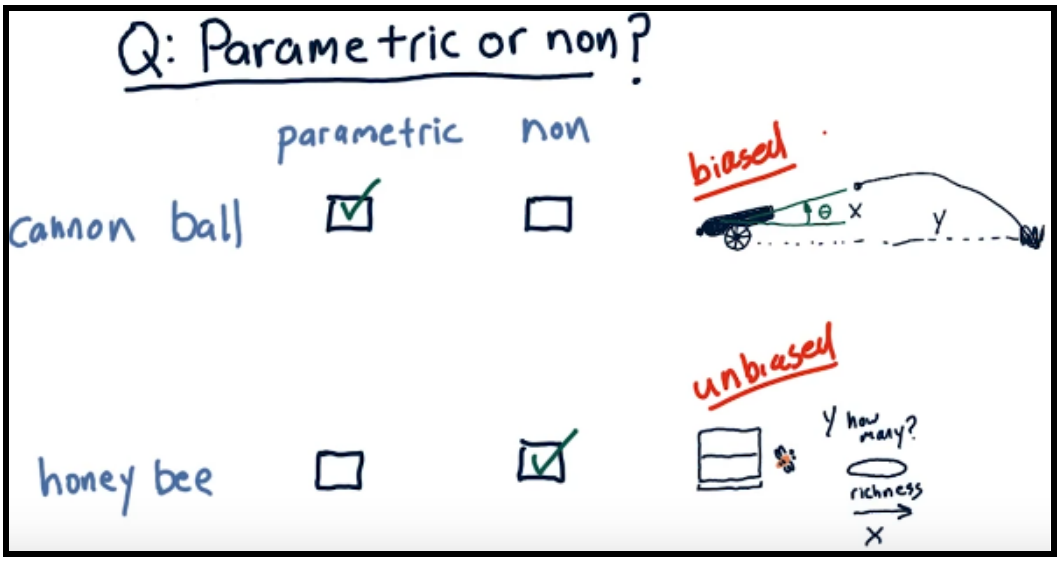


There are a number of methods like these that keep the data around and when they make a query they consult the data to find an answer.
The most famous of these is KNN but there are others such as **kernel regression.**
The main way that KNN differs from kernel regression is that in kernel regression we weigh the contributions of each of the nearest data points according to how distant they are whereas with KNN we weight the contributions equally - each data point that we consider gets essentially an equal weight

Parametric or non-parametric?

- **Yes, the cannon ball distance can be best estimated using a parametric model, as it follows a well-defined trajectory**
    - Because you can start with an equation that will express an estimate of how far a cannon ball will go given this angle
    - What missing are some of the parameters of the equation that reflect the velocity of the cannon ball coming out and so on
    - By taking certain measurements of how far the ball goes at different angles, you can use a parametric learner to find those parameters
    - The key thing is that you can start with an estimate of the underlying behavior of the system in terms of a mathematical equation that expresses how it behaves
    - This is **biased** in the sense that we have an initial guess of what the form of the equation is
        - It makes sense to take advantage of that bias and aim your solution toward that bias
- **On the other hand, the behavior of honey bees can be hard to model mathematically. Therefore, a non-parametric approach would be more suitable**
    - In this case we don't really know, or have a guess of what the underlying mathematical equation might look like and if you don't have a guess, it's better to use a non-parametric or instance based model because it can fit any sort of shape.
    - This solution is **unbiased** because we don't know what the form of the equation is

Quiz



- **Pros**
  - **Parametric**
    - We don’t have to store the original data so it’s very space efficient
    - **Querying is very fast**
  - **Non Parametric (instance based)**
    - New evidence can be added easily since no parameters need to be learned and adding new data points doesn’t consume any additional time so **training is fast**
    - Avoid having to assume a certain type of model, whether it’s linear or quadratic or so on and therefore they’re suitable to fit complex patterns where we don’t really know what the underlying model is like
- **Cons**
  - **Parametric**
    - We can’t easily update the model as more data is gathered - usually we have to do a complete rerun of the learning algorithm to update the model
    - thus for parametric approaches training is slow
  - **Non Parametric (instance based)**
    - We have to store all the data points so it’s hard to apply when we have a huge data set
    - **Querying is potentially slow**

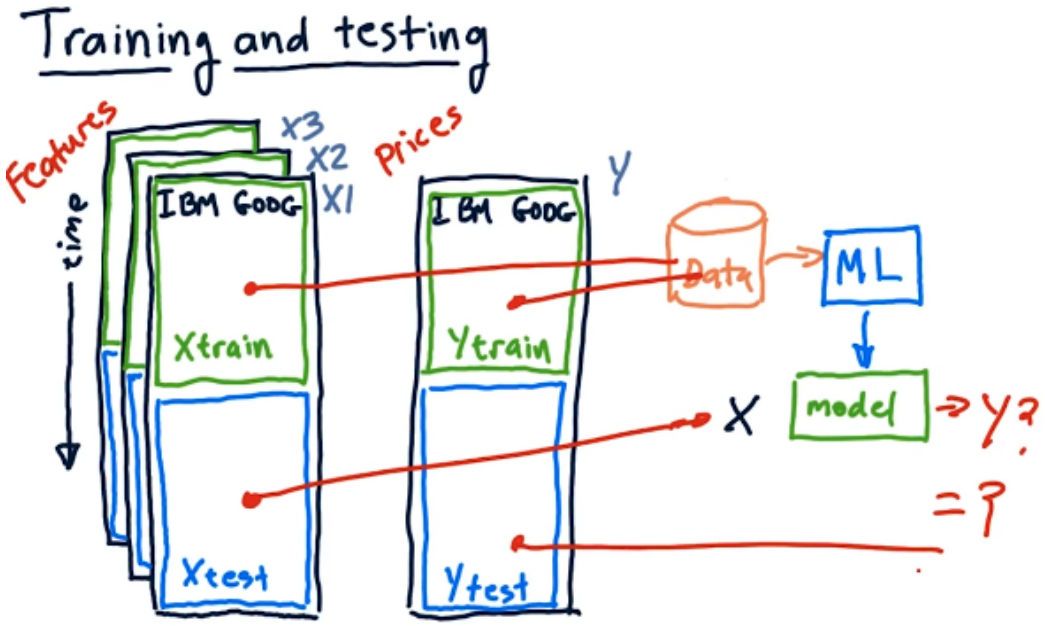
Training and testing

- We're gonna have features that we computed, there are things like Bollinger bands and momentum and price change and things like that. We're going to use those features to try and predict prices or price changes
- We're going to use these features to try and predict prices or price changes

- So this is our X ata and if we've got multiple features we've got multiple dimensions in X - so this might be X1, X2, X3 and so on.
- This other side is our y data which we're trying to predict

In order to evaluate our learning algorithms in a scientific manner, we need to split this data into at least two sections: the training section and the testing section.

- **If we trained over the same data that we tested over, the results would be suspicious because we should obviously be able to do very well if we test over the same data we trained on.** This procedure of separating testing and training data from one another is called**: Out of sample testing.** This is a very important and essential technique.
- We'll call the X data that we use for training **Xtrain** and and the Y data that we use for training **Ytrain** - similarly **Xtest** and **Ytest**.



Learning APIs

For Linear Regression:

- **Learner = LinRegLearner()**
- **learner.train(Xtrain, Ytrain)**
- **Y = learner.query(Xtest)**

For KNN:

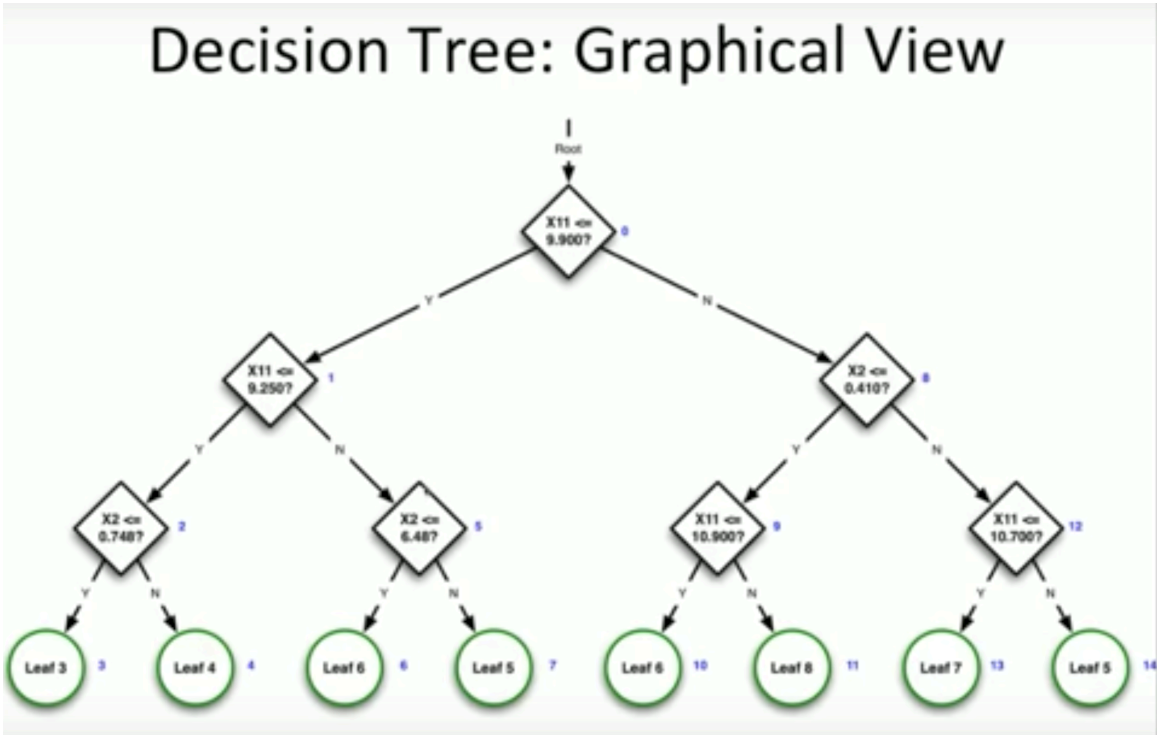
- **learner = KNNLearner(K=3)**
- **learner.train(Xtrain, Ytrain)**
- **Y=learner.query(Xtest)**

```
class LinRegLearner:
        def __init__():
                pass

        def train(X,Y):
                self.m,self.b =favorite_linreg(X,Y)

        def query(X):
                Y = self.m * X + self.b
                return Y
```

The KNN API is going to look exactly the same.

Decision Tree

- They store a tree structure and when a query comes in it essentially bounces down that tree according to factors of the data
- Each node in the tree represents essentially a question: Is this X value greater than or less than this other value?
- Eventually we reach a leaf and that is the regression value that's returned
- Features are also known as factors or columns
- Split value of a node could be selected by using one of the following:
    - Goal: Select the feature which best splits the data into two.
    - Information gain (expensive operations):
        - Entropy: Measures which feature provides maximum split. Once the data is split, it measures the randomness of each group.
        - Correlation: Select feature with the highest correlation to y and split based on that feature.
        - Gini index: Another measure of diversity.
    - Random: Cheap operation. Select random feature and select average of two random rows.
- Each node holds the following:
    - Decision Node:
        - Feature name
        - Split_val
        - Left and right pointer (log base 2)
    - Leaf Node:
        - Leaf node identifier. -1 represents a leaf node.
        - Predicted Y value.



- Training
    - Expensive to build the tree
    - Some features repeat and not all features are used.

    - Most ML implementation try to keep the tree balance in order to guarantee log(n) performance.
- Querying
    - Log(n) performance. Recurse down the tree using the X and return the value of the select leaf.
- Single Random Tree performance very poorly compared to a single Decision Tree with information gain.

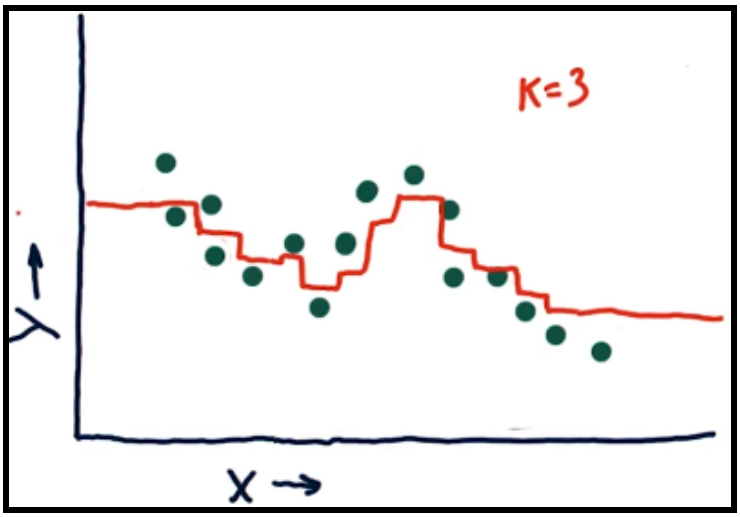
03-03 Assessing a learning algorithm

We've posed the general problem of supervised regression learning and introduced two algorithms that can solve it:

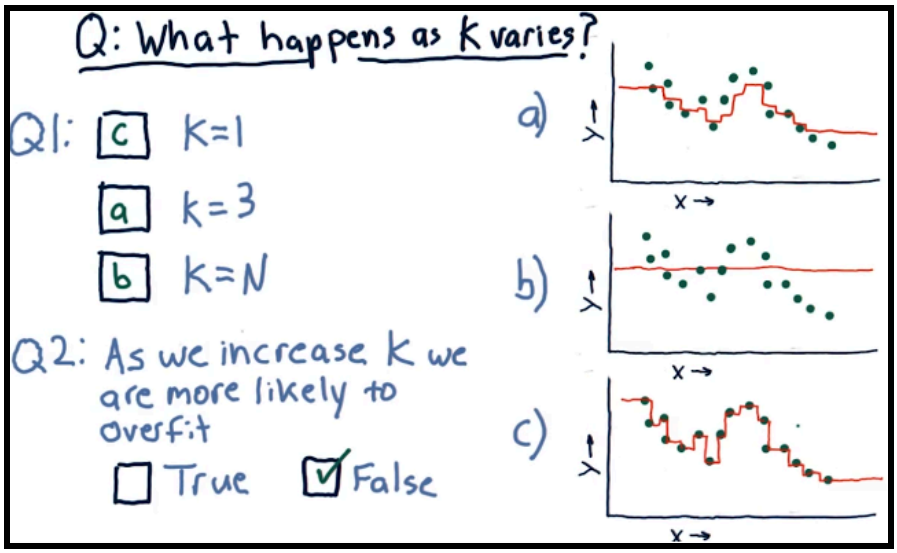
- Linear regression creates parameterized models
- KNN is a non-parametric instance based method

There are in fact many algorithms that can solve this problem and we'll be looking at various methods for assessing those algorithms

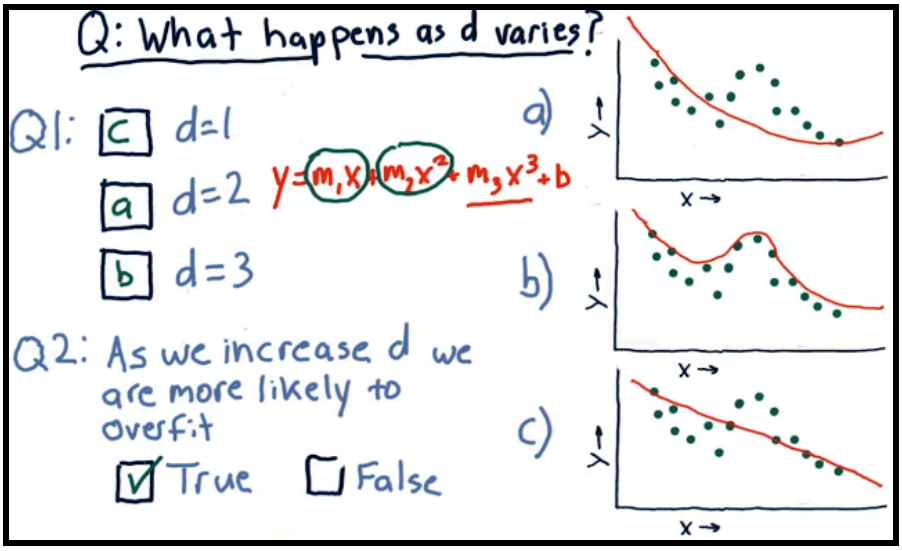
A closer look at KNN solutions

- Green dots are training data
- 

- It doesn't overfit but the model is not very good at extrapolating at the beginning and ends like we might if we had a parametric model



- **As we increase K we are less likely to overfit.**



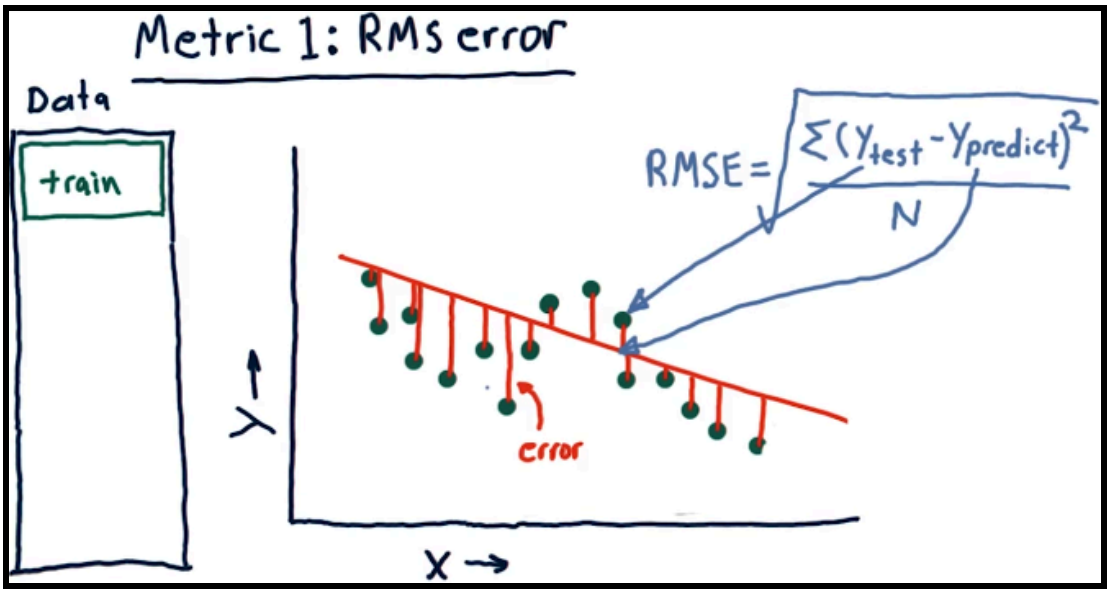
- **As we increase d we are more likely to overfit**
- Yes, in this case, increasing d increases model complexity, and results in our model trying to closely align with the given data points
- As d approaches the number of points, we actually can match the data at every point

NOTE

As we go off the edges for all these models we're able to extrapolate in the direction the data seems to be going and this is a capability that parametric models or these polynomial models have that KNN do not.

Metric 1: Root mean squared (RMS) error

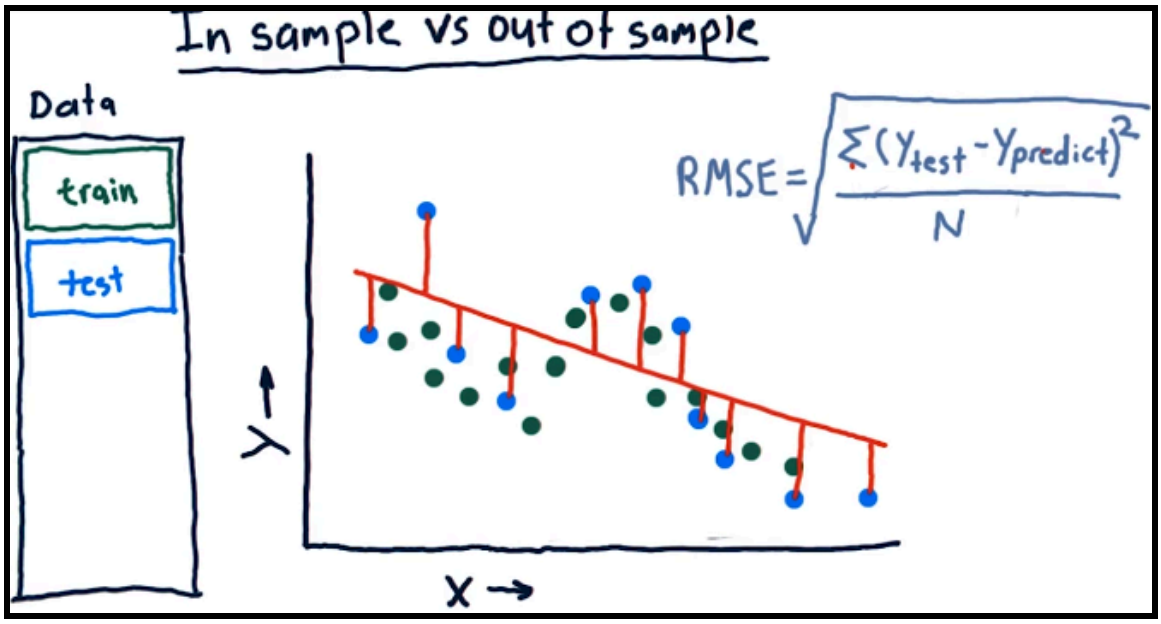
- So far we've seen some graphs that suggest the ways the models can fit the data more or less closely. But let's have a more formal definition of this matching. It's called **error**.
- The standard way to measure error is called RMS error



- We measure against the original training data, but this is arbitrarily small

In sample vs. out of sample

- The more important measure is the out of sample error
- What **out of sample means** is we train on our training set, but we test on a separate testing set of the data



- In this case instead of the green dots we look at the blue dots for our Ytest
- This is our **out of sample root mean squared error.**
- **You would expect out-of-sample error to be larger, since the model has not seen the [blue] points from the test set.**

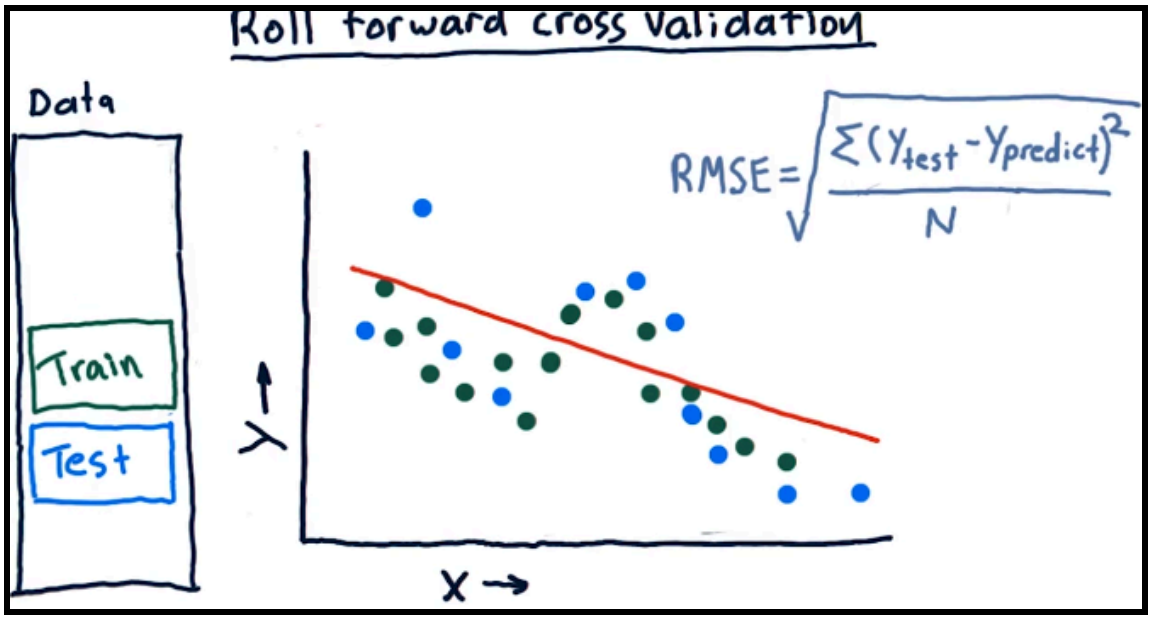
Cross validation

- When researchers are evaluating a training algorithm, they split the data into two chunks. A training chunk (60%) and a test chunk (40%)
- When you don't have enough data you slice it up

- We can get 5 different trials on this one set of data by training on 80% and testing on 20% on any combination of the above and calculating RMS error (RMSE)

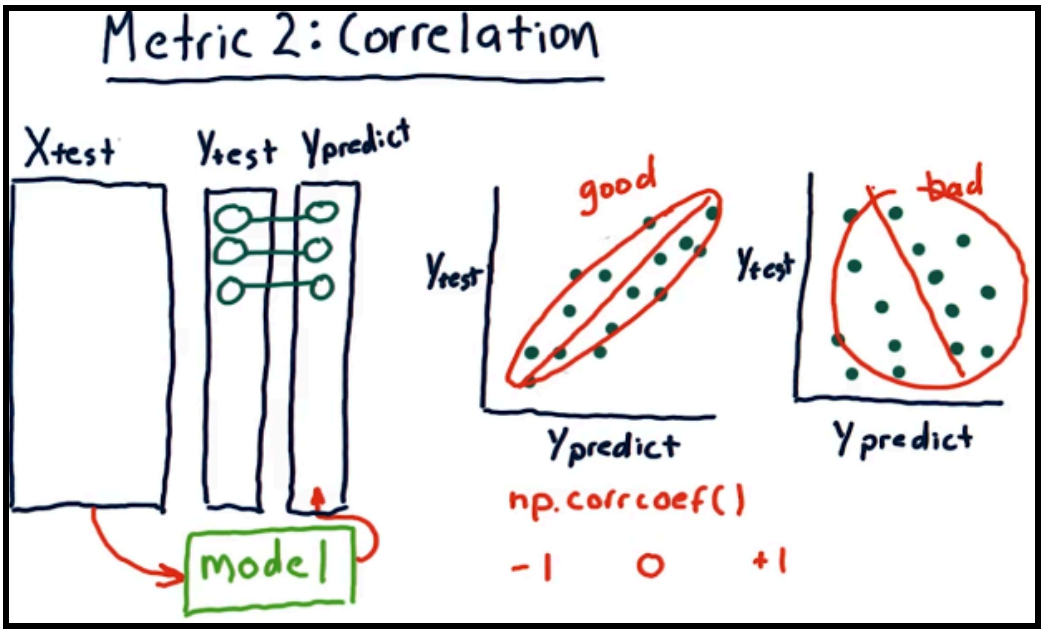
Roll Forward Cross Validation

- Cross validation is a great tool, but the typical usage of it doesn't fit financial data applications well
  - The reason is that it can permit peeking into the future if our training data is after our test data
  - This peeking can lead to unrealistically optimistic results so with this sort of data we need to avoid it. One way to avoid it is with **Roll Forward Cross Validation**
    - That means our training data is always before our testing data, but we can still have multiple trials just by rolling our data forward



Metric 2: Correlation

Another way to visualize and evaluate the accuracy of a regression algorithm is to look at the relationship between predicted and and actual values of our dependent variable Y

- 

    - -1 they're strongly inversely correlated
    - 0 there's no correlation at all
    - 1 they're strongly correlated
- **Correlation is not the slope of the line!**
- **Correlation has to do with how well aligned the points are with the line that we fit**
    - **A nice oval that fits close to that line means it's a high correlation**
    - **A big round thing, we've got poor correlation**

Quiz

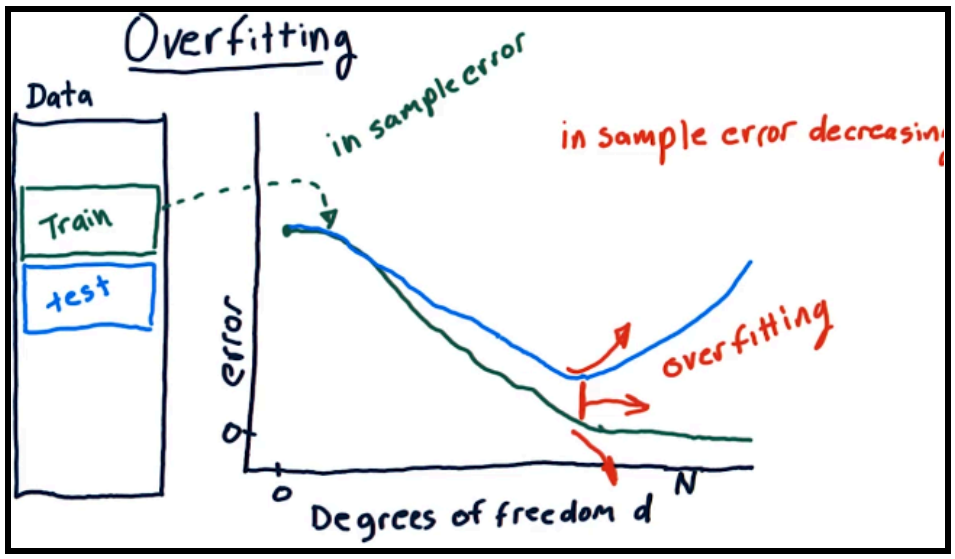
As RMS error increases, correlation decreases.

- **In most cases, as RMS error increases, correlation goes down; however it is possible to construct examples where as RMS error increases, correlation might increase.**
    - **So that also lets you have it correct if you checked that we can't be sure either way**

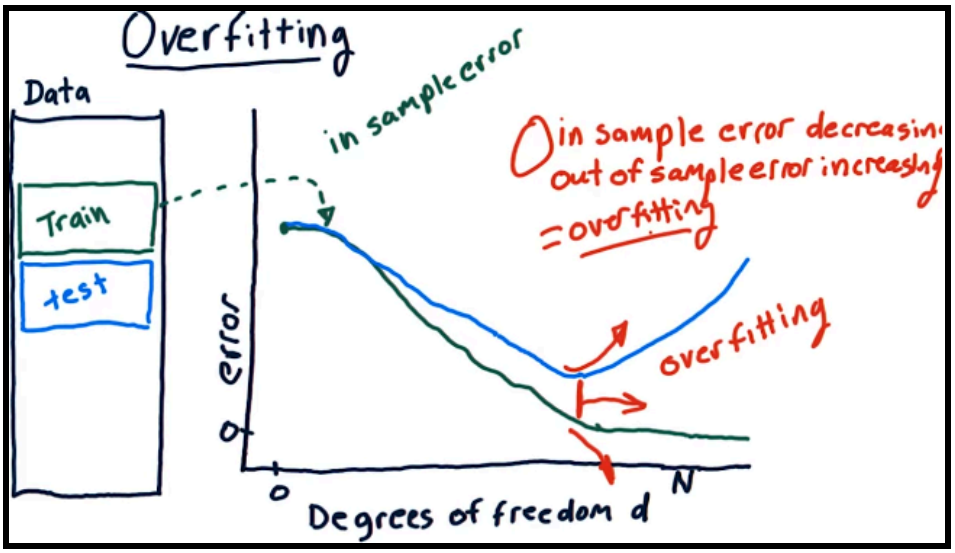
Overfitting

Before we can define overfitting, we need to have a definition of error

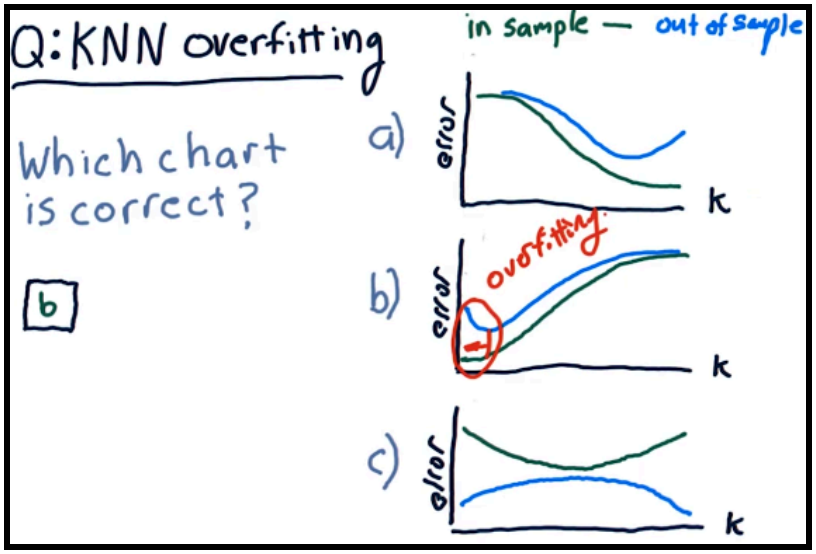
- Let's consider parameterized polynomial models where we can, one at a time, add additional factors like x, x^2, x^3, x^4 and so on
- Let's create a graph where we have along the horizontal axis degrees of freedom, or d the degree of our polynomial and vertically we'll have the error of our model
- Let's measure error as we increase d on our training set
    - **This is in sample error (green line)**
    - **Out of sample error (blue line)**
        - Eventually we reach a point where our out of sample error begins to increase in fact it may increase strongly
        - **In this area, as we increase degrees of freedom, our in sample error is decreasing, but our out of sample error is increasing and that's how we define overfitting!**



- **Overfitting is occurring when the in sample error is decreasing and the out of sample error is increasing**



Quiz



- As k increases, a k-NN approach becomes more and more generalized, since it takes into account more points for each prediction.
- When k = 1, the model fits the training data perfectly, therefore in-sample error is low (ideally, zero). Out-of-sample error can be quite high.
- As k increases, the model becomes more generalized, thus out-of-sample error decreases at the cost of slightly increasing in-sample error.
- After a certain point, the model becomes too general and starts performing worse on both training and test data.
- **As we reduce k down to 1, our in sample error approaches 0 - in fact it becomes a 0 when K = 1 and similarly as we decrease K, our out of sample error decreases, but at some point it begins to increase and that is where we have overfitting. When out of sample error increases and in sample error decreases that is where overfitting occurs.**

Quiz Other considerations

- Space for saving model
  - How much memory do you need in your computer to save the model?
    - **LinReg < KNN (it could be MB or GB of data so KNN is bad)**
- Compute time to train
  - How much compute time do you need to train the model?
    - **LinReg > KNN (it takes 0 time to train KNN, you just add to a data store, with LinReg you have to recompute the factors)**
- Compute time to query
  - How long does it take to query the model?
    - **LinReg < KNN (all you do you plug X in and multiply it out and that's the answer**
- Ease to add new data
  - How easy is it to add new data to your model?
    - **LinReg is harder than KNN (all you have to do is plop it in there, you don't have to recalculate, with LinReg you have to recompute the factors)**

As you can tell, there's always a tradeoff :)

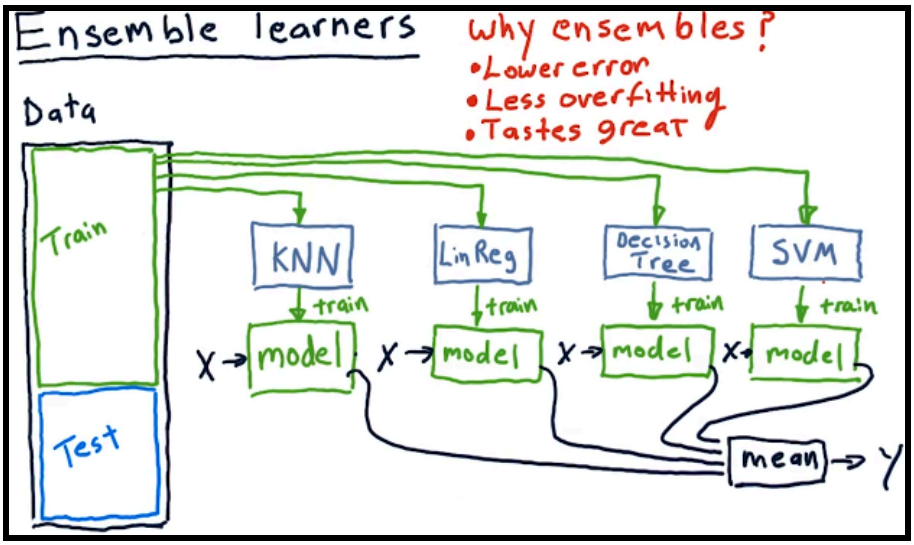
	Training time	Query time	Memory	Bias	Variance	Update data	Normalization
LinReg	mid	low	low	high	low	retrain	
KNN	low	high	high	low	high	n/a	required
DecisionTree	high	mid	mid	low	high	retrain	not required
RandomTree	mid	mid	mid			retrain	

03-04 Ensemble Learning, bagging and boosting

In 1988 Michael Kearns and Leslie Valiant posed the following question:
Can a set of weak learners be combined to create a single strong learner?
One answer to that came in 2009
Back in 2006 Netflix offered a $1M prize for a machine learning algorithm that could do 10% better than their own algorithm at predicting which movies their customers would like to see, the prize was not awarded until three years later in 2009. The winner was **not a single algorithm, but a combination of several - or an ensemble** and this lesson is about **Ensemble learners**

Ensemble Learners

- Creating an ensemble of learners is one way to make the learners you've got better
- We're not talking about creating a new algorithm, but instead assembling together several different algorithms or several different models to create an ensemble learner
- **One thing I want to emphasize here is that you can take what you learn here about ensemble learners and plug it right into what you're already doing with your KNN and linear regression models**

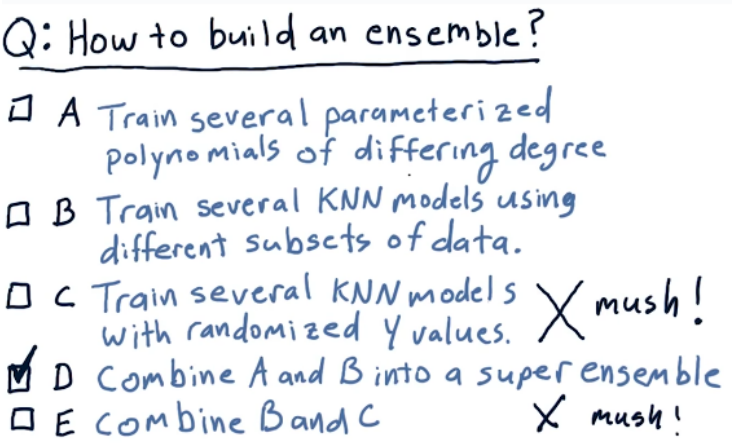


- **We query each model with the same X and we get**

- **Lower error**
- **Less overfitting**
  - The Ensemble of learners typically does not overfit as much as any individual learner by itself. Why is that?
    - As each kind of learner that you might use has a sort of bias - it's easier to talk about that in terms of linear regression in terms of what we mean by bias - so clearly with linear regression our bias is that the data is linear - KNN and Decision Trees have their own. But when you put them together they tend to reduce the individual biases because they all fight against each other in some sort of way.
- **Tastes great**

Quiz - How to build an ensemble?

- **Option C suggests creating kNN models trained on randomized y values - this is not a good idea as the resulting models will essentially be random.**
  - **So, combining B and C won't yield good results.**
- **If we combine several models of different types (here parameterized polynomials and non-parameterized kNN models), we can avoid being biased by one approach.**
- **This typically results in less overfitting, and thus better predictions in the long run, especially on unseen data.**
- **Using randomized Y values is a terrible idea and you get mush!**

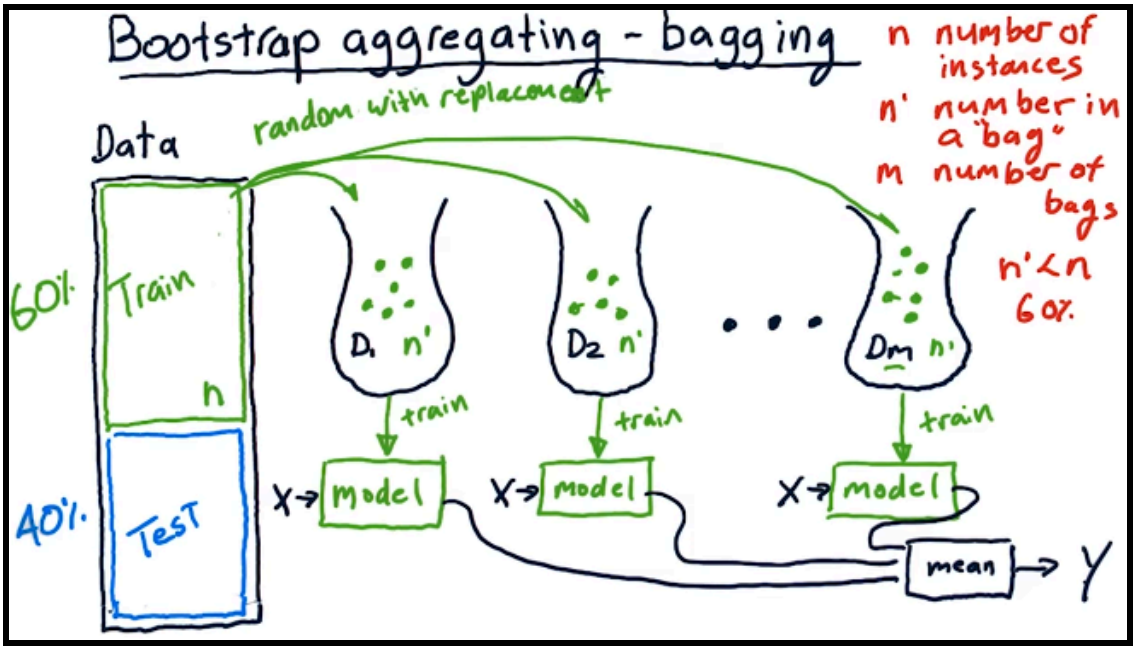


Bootstrap aggregating-bagging

There's another way we can build an ensemble of learners. **We can build them using the same learning algorithm but train each learner on a different set of data, this is called bootstrap aggregating or bagging.**

- It was invented in late 1980's early 90s by Bremen
- We create a number of subsets of the data (bags) and each is a subset of the original data and we collect them randomly with replacement.
- We create M of these groups or bags and each one of them contains n prime different data instances chosen at random with replacement.
- We use each collection or data or bag to train a different model. We have now M different models

We want n' to be < n about 60% which is a good practice

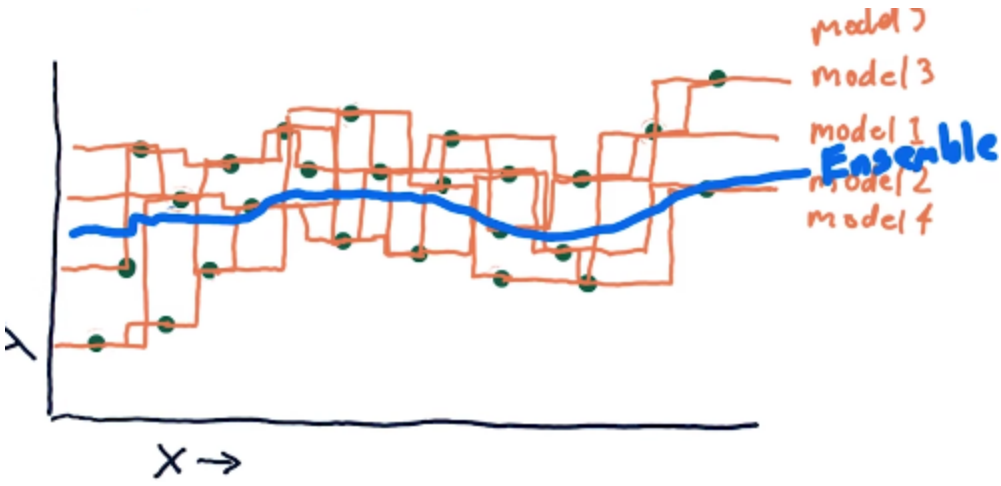


Correction: In the video (around 02:06), the professor mentions that **n'** should be set to about 60% of **n**, the number of training instances. It is more accurate to say that in most implementations, **n' = n**. Because the training data is sampled with replacement, about 60% of the instances in each bag are unique.

Quiz Most Likely to overfit?

- Single 1NN model trained on all the data **(YES)**
  - **Yes, as we saw earlier, a 1NN model (kNN with k = 1) matches the training data exactly, thus overfitting.**
  - **An ensemble of such learners trained on slightly different datasets will at least be able to provide some generalization, and typically less out-of-sample error.**
- Ensemble of 10 1NN learners trained on 60% each (NO)

Bagging Example



Boosting: AdaBoost

Boosting is a fairly simple variation on bagging that strives to improve the learners by focusing on areas where the system is not performing well. One of the most well-known algorithms in this area is called **adaboost** and **ada** stands for **adaptive**.

- We build our first bag of data the usual way
    - We select randomly from our training data
- We then train the model the usual way
- The next thing we do, we take all our training data and use it to test the model
    - In order to discover that some of the points in here, our Xs and Ys are not well predicted
- When we build our next bag of data, we choose randomly from our original data, but each instance is weighted according to the error from the previous bag
    - So points that had significant errors are more likely to get picked and to go into the second bag and any other individual instance
- We build the model of the second bag and then we test the mini ensemble of bag 1 and bag 2 and we measure error again
- Then we build out next bag, our next model by randomly picking but the newest highest erroring ones are more likely to be picked
- We do this over and over again until m or the total number of bags we'll be using (so this is an integer variable)
- **To recap, bagging when we build one of these instance is simply choosing some subset of the data at random with replacement and we create each bag in the same way**
- **Boosting is an add-on to this idea where in subsequent bags we choose those data instances that had been modeled poorly in the overall system before**

Quiz: Overfitting

Which is most likely to overfit as m increases?

- m is the number of bags that we're using or the number of models that we're building to create our ensemble

- Simple Bagging (NO)
- AdaBoost (**YES**)
    - And the reason is that AdaBoost is trying really really hard to match those parts of the data that are off or outliers or whatever, and accordingly it's striving to fit and subsequently it may be susceptible to overfitting
    - **As m increases, AdaBoost tries to assign more and more specific data points to subsequent learners, trying to model all the difficult examples**
    - **Thus, compared to simple bagging, it may result in more overfitting**

Summary: Boosting and Bagging and how it fits in ML4T

- They are just wrappers for existing methods
    - They’re methods for taking existing learners and essentially wrapping them in this meta algorithm that converts your existing learners into an ensemble and you should use the same API to call your ensemble that you would have earlier been using to call an individual learner
        - What this means is that externally to whatever part of your program is calling the learner, it doesn’t know that underneath there you’re doing boosting or bagging
- Your resulting learner is also likely to lower error and reduced overfitage (overfitaje)
- Boosting and bagging are not new algorithms in and of themselves - they’re meta algorithms that let you wrap your underlying learning algorithms into something that’s better.

03-05 Reinforcement learning

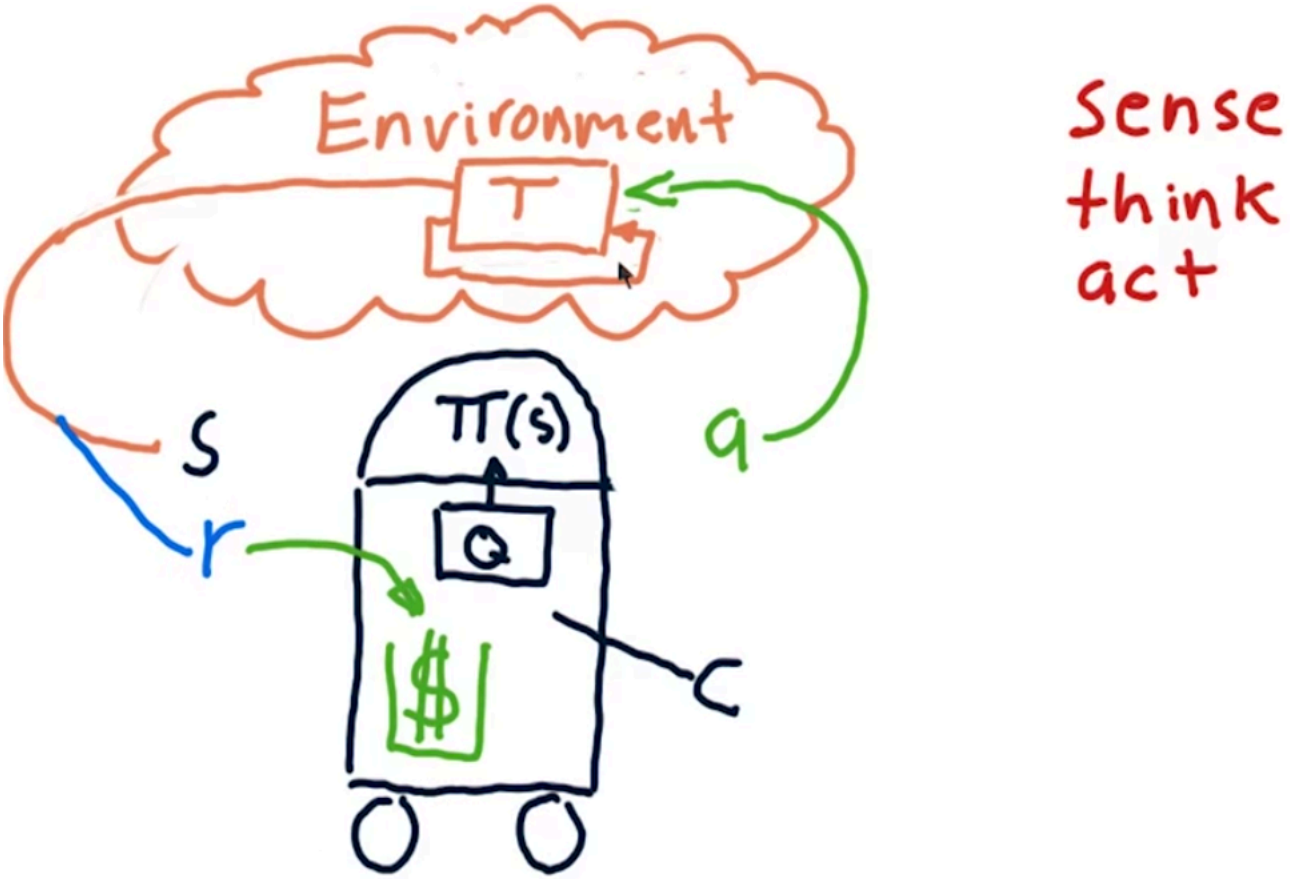
Up to this point we've focused on learners that provide forecast price changes. We then buy or sell the stocks with the most significant predicted price change. This approach ignores some important issues, such as the certainty of the price change. It also doesn't help us know when to exit the position either. In this lesson we'll look at reinforcement learning.

- **Reinforcement Learners create policies that provide specific direction on which action to take**

The RL Problem

- **It's important to point out that when we say reinforcement learning we're really describing a problem, not a solution**
    - In the same way that linear regression is one solution to the supervised regression problem, there are many algorithms that solve the RL problem

For a robot:



- Our robot is going to interact with the environment
    - The sense, think, act cycle and you don't have to implement it only using reinforcement learning
    - There's many ways that you could implement sense, think act but we're going to focus on how to do that with reinforcement learning
- Our robot observes the environment and some form of description of the environment comes in:
    - Let's call that the state s, so s is our letter that represents what we see in the environment
- Now the robot has to process that state somehow to determine what to do
    - We call this π (pi), or policy
        - So the robot takes in the state s and then outputs an action **a = π(s)**

- Action **a** affects the environment in some way and it changes it
    - Now this is a sort of circular process
        - The action **a** is taken into the environment and the environment then transitions to a new state, so **T** is this transition function that takes in what its previous state was and the action and moves to a new state
        - And that new state comes out boom, back into the robot
        - Robot looks at his policy
        - Action comes out

The question is, how do we arrive at this policy?

- The above example (puzzle) is missing a piece and that's the thing that helps us find pi - and that piece is called **r**, which is the reward
- So every time the robot is in a particular state and it takes an action
- There's a particular reward associated with taking that action in that state and that reward comes into the robot - and you can think of the robot having a little pocket where it keeps cash and that's what reward is.
- The robot's objective, over time is to take actions that maximize this reward
    - Somewhere within the robot there's an algorithm that takes all this information over time to figure out what that policy ought to be

Recap

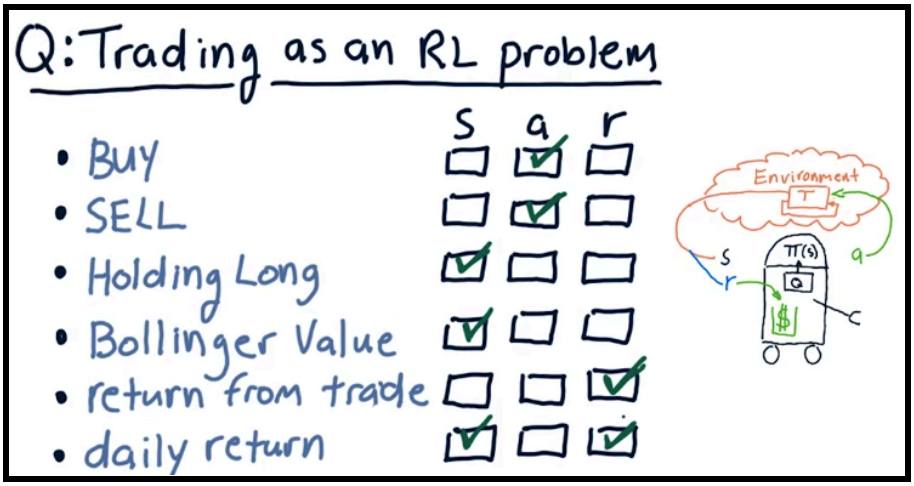
- **S** is the state of our environment and that's what the robot sense in order to decide what to do. It uses this policy pi to figure out what that action should be
- Pi can be a simple lookup table
- Over time, each time the robot takes an action, it gets a reward and it's trying to find the pi that will maximize reward over time
- **In terms of trading, our environment really is the market and our actions are actions we can take in the market like buying and selling or holding**
- **S are factors about our stocks that we might observe and know about**
- **R is the return we get for making the proper trades**

Quiz: Trading as an RL problem

As you know we want to use RL algorithms to trade with - so let's think about how we can map the trading problem to reinforcement learning

- **Is that item a description of our state that we ought to consider before we make a trade?**
- **Is it an action that we give to the market to cause a trade to occur?**
- **Or is it a potential reward that we would use to inform our algorithm for learning how to trade?**
- Is there a potential for some of these to serve more than one role?

Think about which ones are actions (things the trading agent can do), which represent states - either of the stock market or one's portfolio, and which ones come as rewards as a result of taking an action.



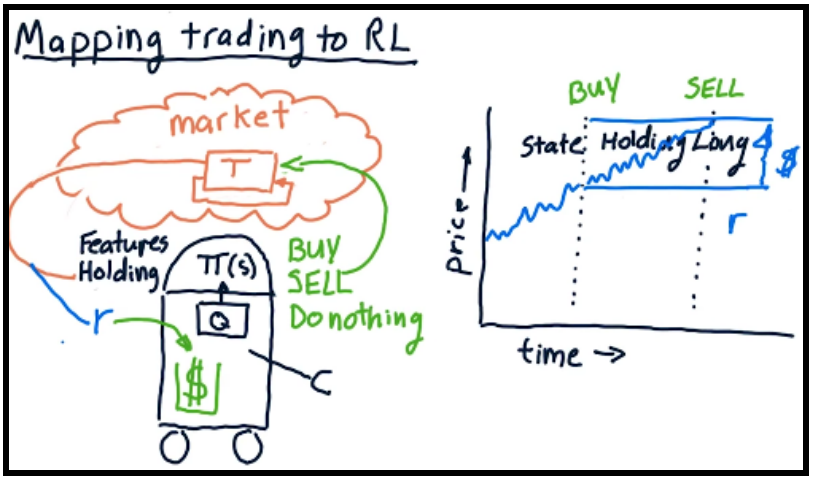
- **BUY, SELL** - are actions, those are directives we give to the market or the environment to change it and potentially change our state
- **Holding Long** is a part of the state, it tells us whether we are holding the stock or not
    - We might also be holding short if we had shorted the stock
- **Bollinger Value,** that's a feature, a factor that we can measure about a stock and that's part of the state as well - that would inform us whether we wanted to act on it in some way with an action
- **Return from trade** - when we finally exit a position - that is our reward
    - We might lose money, so it would be a negative reward if we lost money
    - We might make money and that'd be a positive reward
- **Daily Return** - that could be either a state, in other words a factor we consider for deciding what to do, but it could also be a reward we'll get into that more later and you'll see how it could be one or the other

Mapping trading to RL

Let's consider now, a little more carefully how we map trading to an RL problem

- First of all the environment here is really the market
- Our state that we're going to consider includes things like
    - market features
    - Prices
    - whether we're holding the stock
- Our actions are things like BUY and SELL and potentially do nothing is also an allowable action

So let's think about this in the context of trying to learn how to trade a particular stock



- So we've got this historical time series and let's say this left vertical dotted line is today

- Now we can look back over time to infer the state of the stock
    - So what are the Bollinger Band values and things like that
- Then we process the historical data and decide what's our action
    - Let's suppose that we decide to buy
        - Once we buy, we're now holding long - that's part of our state
        - We go forward, we're now on a new state where the price has gone up, we're holding long and let's suppose we decide to sell at that point
    - So we've had two actions, well we've been in two states - in state one we were not holding - we executed the action BUY - then we went forward in time we're holding long now and then we execute the action sell
    - Note that we made money here at the second vertical dotted line and that's our reward r
    - The policy that we learn tells us what to do each time we evaluate state and we're going to learn that we haven't talked yet about how we learned the policy but we're going to learn the policy by looking at how we accrue money or don't based on the actions we take in the environment

Markov Decision Problems

Let's formalize things a little bit

What we've been working with is something called a **Markov Decision Problem (MDP)**

- Set of states **S**
    - These are all the values that this S can take as it comes into the robot
- Set of actions **A**
    - These potential actions we can take to act on the environment
- Transition function **T[s,a,s']**
    - The T within the environment - it's a 3D object and it records in each of its cells the probability that if we are in state **s** and we take action **a**, we will end up in state **s'**
        - Something to note about this transition function is, suppose we're in a particular state **s** and we take a particular action **a** - the sum of all the next states we might end up in has to sum up to one
        - In other words, with probability one, we're going to end up in some new state, but the distribution of probabilities across these different states is what makes this informative and revealing
- Reward function **R[s,a]**
    - An important component that gives us the reward if we're in a particular state and we take an action a we get a particular reward
- **So if we have all these things defined we have what's called a Markov Decision Problem MDP**
- Now the problem for a reinforcement learning algorithm is to find this policy **pi(s)** π**(s)** that will maximize reward over time and in fact if it finds the **optimal** policy we give it a little symbol pi* π^* to indicate that is optimal
- **If we have these and in particular, if we have T and R - there are algorithms we can unleash that will find this optimal policy**
    - **Two of them are:**
        - **Value Iteration**
        - **Policy Iteration**

Since we don't start off knowing T and R and so we're not going to be able to use these algorithms directly to find this policy

Unknown Transitions and Rewards

- Most of the time we don't have this transition function and we don't have the reward function either.
- So the robot or the trader, whatever environment we're in, has to interact with the world, observe what happens, and work with that data to try to build a policy
    - Experience tuple: **<s1,a1,s'1,r1>**
        - Here we're saying when you observe the state **s1**, you take action **a1** you end up in this new state at least it's an example of you ending up in this state **s1'** and reward **r1**
            - Which is very similar to experience tuples in regression learning where we have an **X** and a **Y** paired together, so when you observe this X, you see this Y
        - Now we find ourselves in this new state **s2** which is really **s1'** we take some new action **a2** and we end up in some new state **s2'** and we get a new reward **r2**
        - **We do this over and over again gathering experience tuples all along the way**
- Now if we have this trail of experience tuples there's two things we can do with them in order to find that policy **pi**
    - A set of approaches is called **Model Based** reinforcement learning
        - What we do is look at this data over time and we build a model of T just by looking statistically at these transitions
            - In other words we can look at every time we were in a particular state and took a particular action and see which new states we ended up in and just build a tabular representation of that
        - Similarly we can build a model of R
            - We just look statistically when we're in a particular state and we take an action, what's the reward? We can just average that over all these instances
        - Once we have these models, the T and R we can then use value iteration or policy iteration to solve the problem
    - Another set of approaches is called **Model Free** and that's the type we're going to focus on - in particular we're going to learn about **Q-learning**
        - **Model Free** methods develop a policy directly by looking at the data and of course we'll talk about those soon

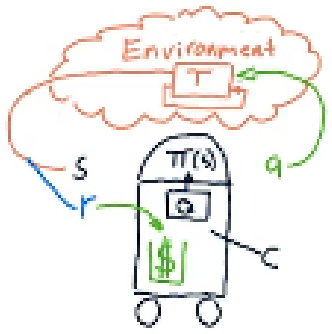
Unknown transitions and rewards

Experience tuple

- Model-based
  - Build model of
    - T[s,a,s']
    - R[s,a]
  - Value/policy iteration
- Model-free
  - Q-Learning.

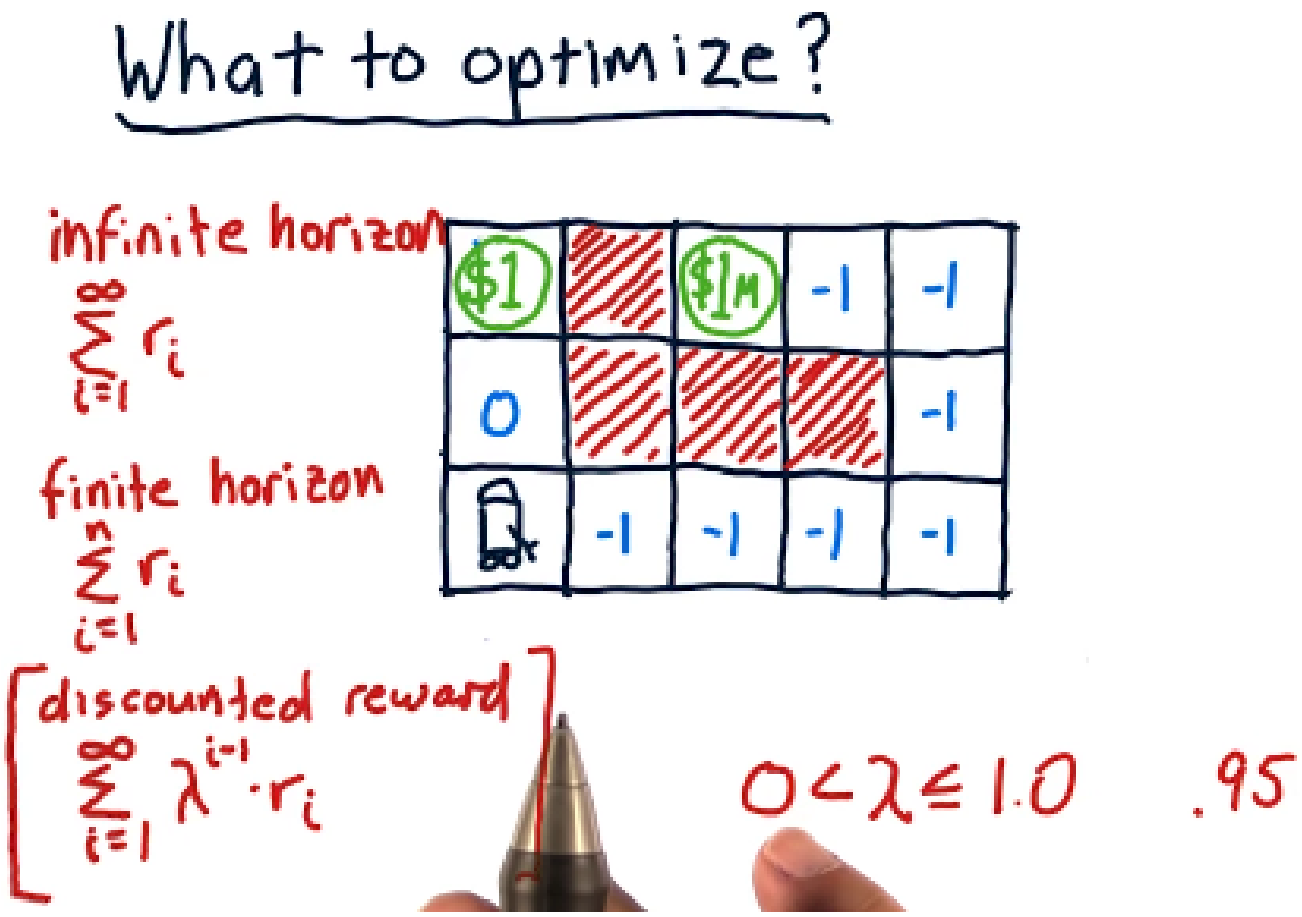
$\langle s_1, a_1, s_1', r_1 \rangle$

$\langle s_2, a_2, s_2', r_2 \rangle$



What to optimize?

- One dollar a year for a million year lottery winner! One dollar delivered to us in a million years in the future is really not as valuable as a dollar or that we get now
- If we think about a robot living forever it might do something just mundane to gather a dollar a year - that's an infinite amount of money but in practice it doesn't really work that well
  - So to consider that and to illustrate that we're going to see a little maze problem and we'll think about what the robot ought to do that would be optimal in this maze



- Here's our robot and the challenge for our robot
- We have a reward of $1 and a reward of $1M
    - So if the robot comes over and gets the $1, it's special in that each time he touches it, he gets $1 and it goes away but then it comes back
        - So it could go back and forth between 0 and $1 each time it moves
    - For $1M, once the robot tags it it's gone but clearly it's worthwhile to come over here and grab it
- Red are obstacles and it can't go there
- Blue are rewards or penalties that the robot gets as it enters a state

Now if we say that what we want to optimize is the sum of all future rewards, then it doesn't matter whether we go this way and just get that dollar over and over and over again or we go get the million, then come back and get the one dollar over and over again.

- There is no difference because they both sum to infinite over time

Now what if we say, okay, i want to optimize my reward over three moves - so we have a finite horizon

- If we go for the $1M the finite horizon reward is $-3
- If we go for the $1, the finite horizon reward is 1

If we optimize over 8 moves, we get the $1M or we get $3

So clearly as we expand our finite horizon, trivially up to say eight steps going this way and tagging at one million is the best thing to do

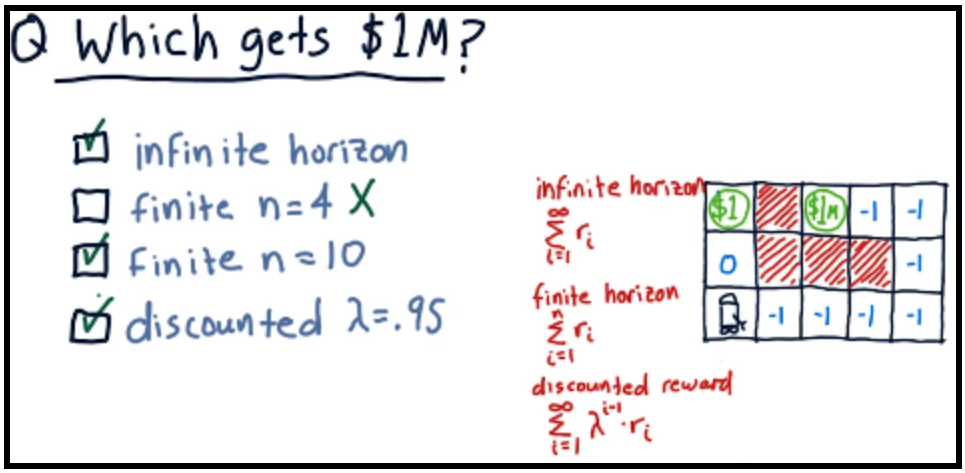
If we carried it even further, we'd discover that then we should come back and go to the $1 over and over again

- **Formally with the infinite horizon, what we're trying to maximize is the sum of all rewards over all of the future**
    - So it's the sum of each of these rewards for i from 1 to inf
- **The finite horizon is very similar, it's just we don't go to infinity**
    - So for optimizing over a horizon of four steps, n would be 4 so i goes from 1 to 4 - we're just trying to maximize the sum of the reward for the next 4 steps

- **There is yet one more formulation that if you think back to that lecture a while back about what's the value of a future dollar we can dig that up and it makes a lot of sense in terms of RL**
    - So remember that if it takes us say, four years to get a dollar that dollar is less valuable than say if it takes one year
        - In the same way, if it takes say, eight steps to make a dollar - that dollar is less valuable than a dollar I can get just in one step and the way we represent that is very simple just like we represented the sum of future dividends and it looks like this: **It's called discounted reward**
            - **So instead of just summing up the ri, we multiply it by this factor gamma^i-1 such that our immediate reward, the very next one we get, whatever gamma is when it gets raised to the zero power is just one, so for the very next step we get r but for the step after it, it's gamma^1 so it devalues that reward a little by by a factor of gamma^1**
                - Gamma is a value between 0 and 1
                    - The closer it is to 1, the more we value rewards in the future
                    - The closer it is to 0, the less we value rewards in the future
                - In fact if gamma is set equal to 1, the formula is exactly the same as the infinite horizon formula (**both are in the figure above**)
                - **But gamma relates very strongly to interest rates if you recall**
                    - Let's say gamma were 0.95, it means each step in the future is worth about 5% less than the immediate reward if we got it right away
            - **The discounted reward is the method that we use in Q learning**
                - The math turns out to be very handy and it provides nice conversion properties

Q Which gets $1M?

- Infinite Horizon
    - Is a little iffy because the robot can go this way and get a dollar on every other move and that will add up to infinity
    - It can go to $1M and then come back and do the infinity again
- Finite n=4
    - It won't get to $1M because it will either only get negative rewards for 4 steps or just $1 for 2
- Finite n=10
    - Will reach $1M
- Discounted gamma=0.95
    - Where each dollar in the future is only worth 0.95 and that gets smaller as we get further and further into the future but by the time we get to the 8 steps that it takes to reach this $1M reward, it's still so huge that that's clearly the optimal thing to do
    - ***Both a finite horizon of n = 10 as well as discounted rewards will result in the robot picking a path to the $1M cell.***
    - ***Infinite horizon may also lead to the robot choosing the $1M cell, but there isn't much difference mathematically since repeatedly visiting the $1 will also result in infinite reward.***



RL Summary

- RL algos solve MDPs
  - The problem for reinforcement learning algorithms is a Markov Decision Problem and RL solve them
- A MDP is defined by State, Actions, Transition, Rewards -> Policy
  - T[s,a,s'] - what is the probably that being in state s and taking action a you will end up in s'
- The goal for reinforcement learning is to find a Policy PI that maps a state to an action that we should take and its goal is to find this pi such that it maximizes some future sum of the reward
  - We talked about that being either infinite horizon
  - Fixed horizon
  - Or Discounted sum
- Map trading to RL
  - S are states, features about stocks and whether or not we're holding a stock
  - Actions are BUY, SELL, or do NOTHING
  - The Transition function here is the market
  - The Reward function is how much money we get at the end of a trade
  - So we can apply RL algorithms to find this policy
    - Value Iteration
    - Policy Iteration
    - Q Learning

03-06 Q-Learning

- Q-Learning is a model-free approach, meaning that it does not know about or use models of the transitions T or the rewards R
- Instead Q-learning builds a table of utility values as the agent interacts with the world
- These Q-values can be used at each step to select the best action based on what it has learned so far
- **The fantastic thing about Q-learning is that it is guaranteed to provide an optimal policy**
    - There is a hitch however, that we'll cover later

What is Q?

Q-learning is named after the Q function.

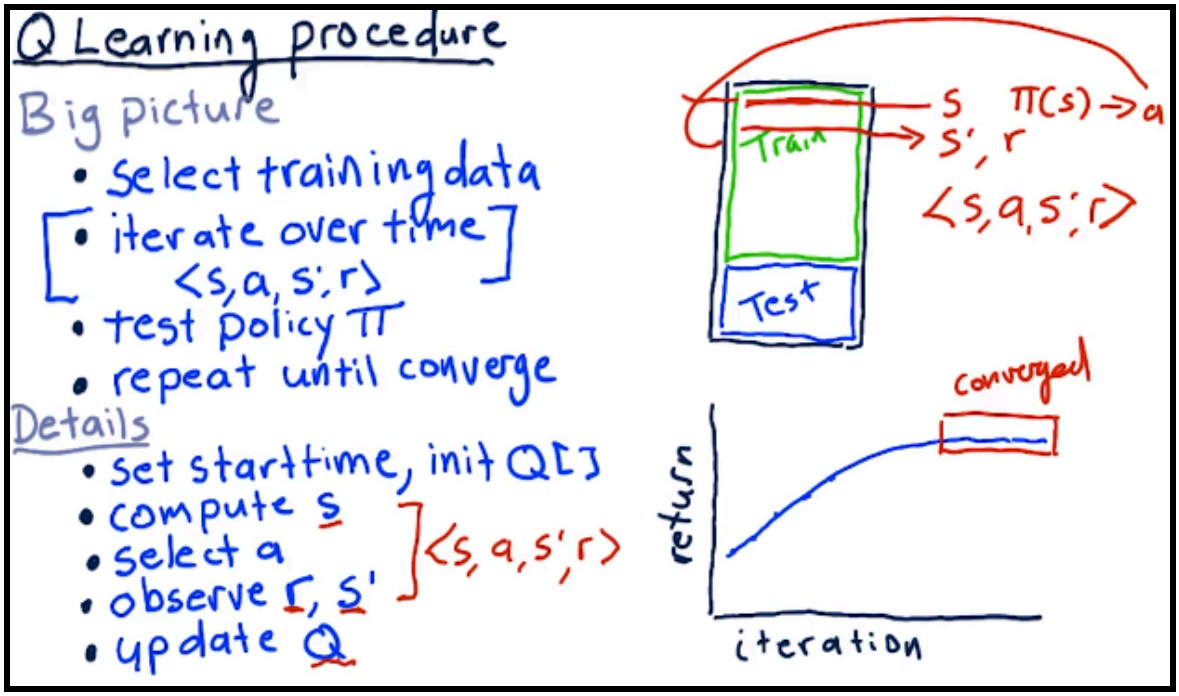
- Q can be written as a function or it can be written as a table
    - **In this class we're going to view Q as a table which has two dimensions**
- s is the state we're looking at and a is the action we might take
- Q represents the value of taking action a in state s and there's two components to that
    - The two components are the **immediate reward** you get for taking action a in state s plus the **discounted reward -** which is the reward you get for future actions
    - An important thing to know is that Q is not greedy in the sense that it just represents the reward you get for acting now, it also represents the reward you get for acting in the future
    - **Q[s,a] = immediate reward + discounted**

How to use Q?

Let's suppose we have Q already created for us, we have this table - how can we use it to figure out what to do?

- So what we do in any particular state is the policy and we represent the policy with pi π **:**
  - **π(s) = argmax_a(Q[s,a])**
    - Means what is the action we take when we are in state s, or what is the policy for state s and we take advantage of our Q table to figure that out
    - So we're in state s and we want to find out which action is the best
      - All we need to do is look across all the potential actions and find out which value of Q[s,a] is maximized - so we don't change s, we just step through each value of a and the one that is the largest is the action we should take
    - **The mathematical way to represent this is to use the function argmax**
      - It find the a that maximizes Q[s,a] and the answer is **a**
- After we run Q-learning for long enough we will eventually converge to the optimal policy **π*(s) pi*** and the optimal Q table is **Q*[s,a]**
  - **This is how you use a Q table if you have it, and next we consider how do we build the Q table in the first place**

Q-learning procedure



The big picture at a high level of how we train a Q learner:

- **Select training data**
  - We have our data and we select the data we want to train on
    - In the case of the stock market it's time series so it's arranged from oldest to newest vertically in the figure
- **Iterate over time <s,a,s',r>**
  - We evaluate the situation at the red line for a particular stock that gives us s our state

    - We consult the policy that gives us an action pi(s) -> a
    - We take the action and plug it into the system at the second red line and we evaluate the next state and we get our s' and our reward
    - **After one iteration we've got s, a, s' and r or an experience tuple**
        - We use this experience tuple to update our Q table
    - **::In more details::**
        - **set startime, init Q[]**
            - set start time
            - we initialize our Q table Q[] with small random numbers and variations to this are fine
        - **compute s**
            - We observe the features of our stock or stocks and from those build up together our state s
        - **select a**
            - We consult our policy, or in other words we consult Q to find the best action in the current state that gives us a
        - **observe r, s'**
            - Then we step forward and we see what reward we get and what's our new state
        - **update Q**
            - We now have a complete experience tuple that we can use to update our Q table
            - So we take this information that we just learned and we improve Q based on that information
        - **Then we step to the next point in time and the next and the next and so on**
- **Test policy pi π**
    - Once we get through all our training data, we test our policy and we see how well it performs in a back test
- **Repeat until convergence**
    - If it's converged or it's not getting any better, we say we're done, if not, we repeat this whole process all the way through the training data
    - What do we mean by converge?
        - Each time we cycle through the data training our Q table and then testing back across that same data, we get some performance
        - We expect that each time we complete an iteration here, our performance is going to get better and better, but after a point it finally stops getting better and it converges
        - So we eventually reach a regime where more iterations doesn't make it better and we call it converged at that point like in the figure

Update Rule

- Consider our robot here that's interacting with the world
- The first thing it sees is some state
    - The thing to do now is to go look in the Q table and find the action that corresponds to the maximum Q value and then take that action
- Once that action is taken, that results in two new things
    - One is a new state s'
    - And the other is a reward r
- All this information comes into the robot and it needs to use that information to update its Q table

- So as a consequence of this interaction with the world, it's got a s, an a, a s' and a r

How does this information improve the Q table?

- There are two main parts to the update rule:
  - The first is what is the old value that we used to have? And that's Q[s,a] and what is our improved estimate? And we want to blend them together
    - To combine/blend them we introduce this new variable called alpha which is the **learning rate** and which can take on any value from 0 to 1 and usually about 0.2
      - What this means is that our new improved version of Q, Q' is a blend of alpha times the improved estimate plus 1 minus alpha of the old value
    - Larger values of alpha causes us to learn more quickly, lower values of alpha cause the learning to be more slow
      - **So a low value of alpha means that in the update rule, the previous value for Q[s,a] is more strongly preserved**
  - Now we also introduce gamma which is the discount rate
    - **A low value of gamma means that we value later rewards less**
    - Same thing about the discount rate when we talk about bonds - a low value of gamma equates to essentially a high discount rate
    - **A high value of gamma, near 1 means that we value later rewards very significantly**
      - If we were to use 1.0, that means a reward in 20 steps in the future is worth just as much as a reward right now

- 

represents our future discounted rewards, in other words we end up in state s' and from then on out we're going to act optimally or at least the best that we know how to and the question is:
  - **What is the value of those future rewards if we reach state s' and we act appropriately?**
    - It is simply that Q value!
      - But we have to find out what the action is that we would have taken so that we can reference the Q table properly
  - So if we're in state s', the action that we would take that would maximize our future reward is argmax a' where a' is the next action that we're going to take with regard to Q[s',a'] so we're going to find that best a' that best action that maximizes the value when we're in that state.
  - So this collapses just to a' and then we look at the Q table value for Q[s',a'] which allows us to bring it all together
- The update rule is now the following
  - **Our new Q value in state s action a Q[s,a] is that old value, multiplied by 1-alpha, so depending on how large alpha is, we value that old value more or less, plus alpha times our new best estimate and our new best estimate is again, our immediate reward r plus the discounted reward for all of our future actions**

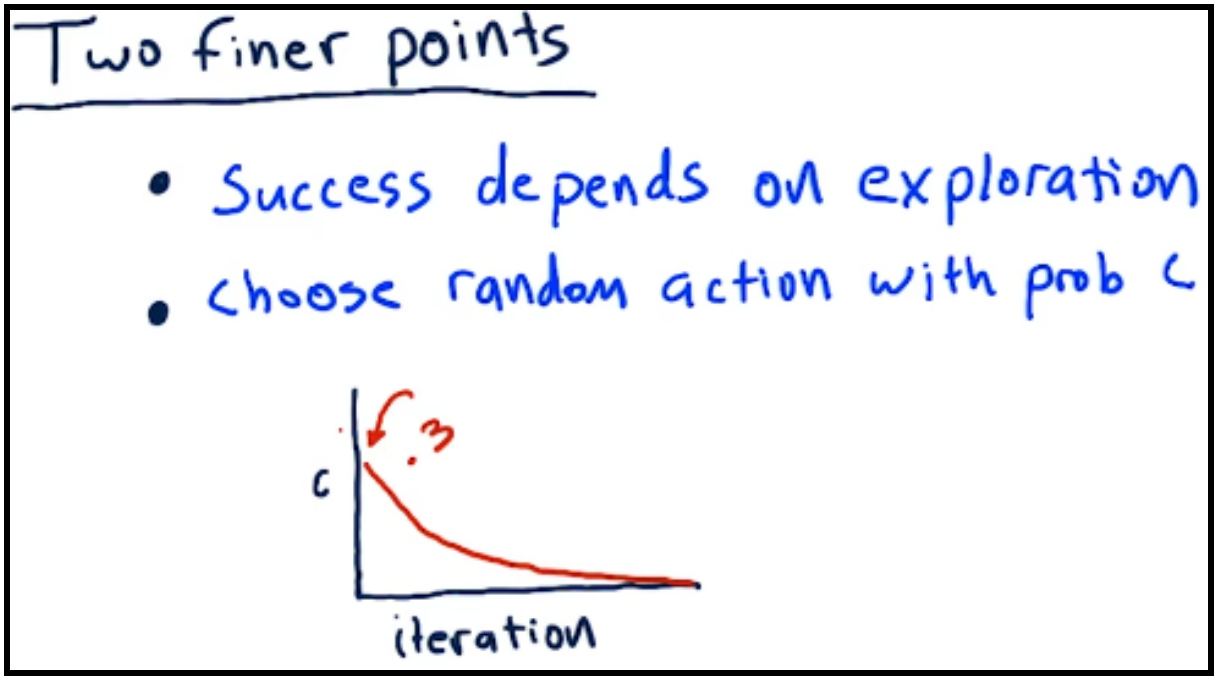
The formula for computing Q for any state-action pair <s, a>, given an experience tuple <s, a, s', r>, is:

Q'[s, a] = (1 - α) · Q[s, a] + α · (r + γ · Q[s', argmax_a'(Q[s', a'])])

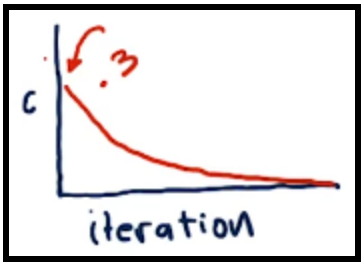
Here:

- r = R[s, a] is the immediate reward for taking action a in state s,
- γ ∈ [0, 1] (gamma) is the *discount factor* used to progressively reduce the value of future rewards,
- s' is the resulting next state,
- argmaxa'(Q[s', a']) is the action that maximizes the Q-value among all possible actions a' from s', and,
- α ∈ [0, 1] (alpha) is the *learning rate* used to vary the weight given to new experiences compared with past Q-values.

Two finer points



- Success depends on exploration
  - The success of Q-learning depends to a large extent on exploration so we need to explore as much of the state and action space as possible
    -
- Choose random action with probability c
  - One way to accomplish this is with randomness and the way we can interject that fairly easily in the step or Q-learning where we are selecting an action, we flip a coin and randomly decide if we're going to randomly choose an action
    - which is really two flips of the coin:
      - The first flip:
        - Are we going to choose a random action?
          - **If we decide OK, we're going to choose an action randomly** we now need to **flip the coin a second time** again to choose which of those actions we're going to select
        - Are we just going to pick the action with the highest Q value?
  - **A typical way to implement this randomness is to set the probability at about 0.3 or something at the beginning of learning and then over each iteration to slowly make it smaller and smaller and smaller until we essentially don't choose random actions at all**

- 

    - What we gain here is when we're choosing these random actions fairly frequently is we're forcing the system to explore and try different actions in different states
    - It also causes us to arrive at different states that we might not otherwise arrive at if we didn't try those random actions

The trading problem: Actions

Now that we have a basic understanding of Q learning, let's consider how we can turn the stock trading problem into a problem that Q learning can solve.
To do that, we need to define our actions, states and rewards.

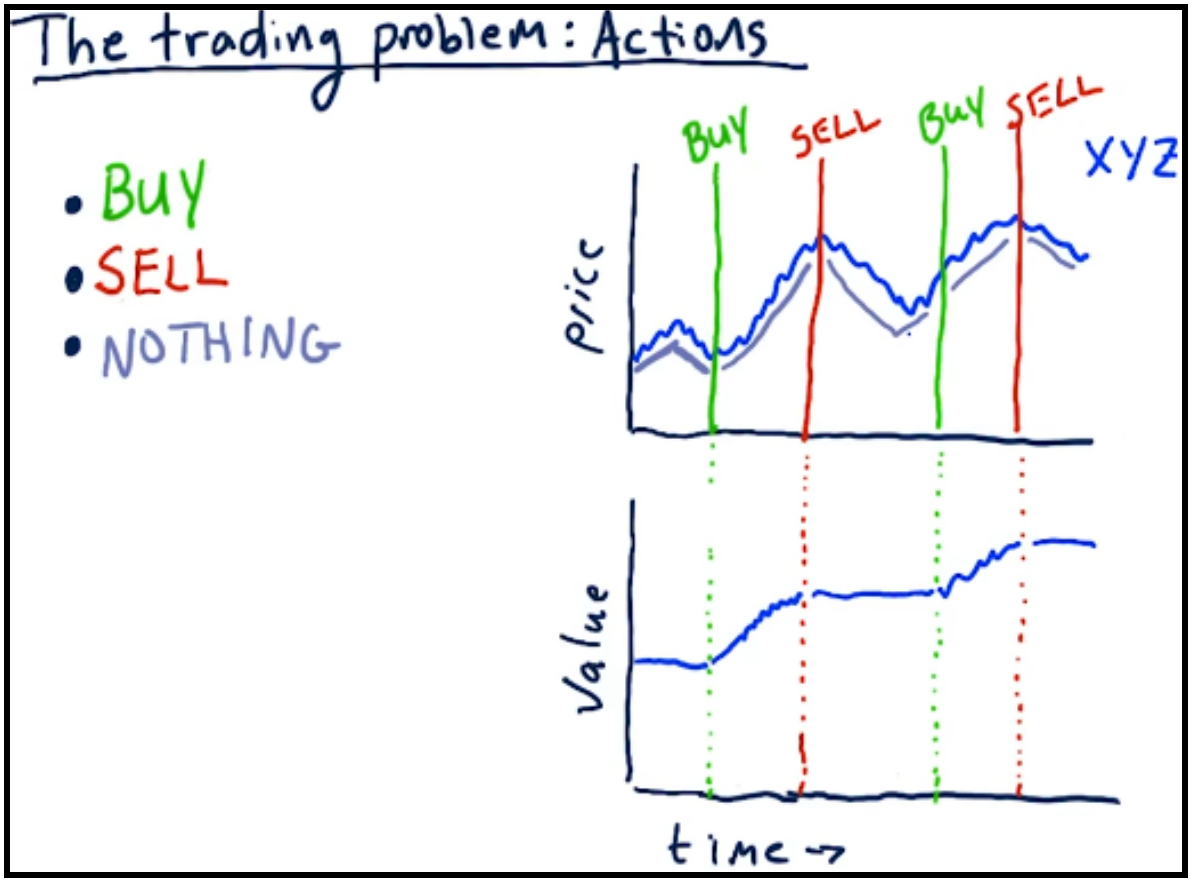
Actions

- **BUY**
- **SELL**
- **NOTHING**
    - Most frequently we do nothing

1. So we're evaluating the factors of the stock, we compute several technical indicators which is our state, we consider our state and we do nothing
2. Let's suppose we do nothing for quite a long period here, but after a while, boom - something triggers and says we should buy.
3. So we buy the stock, we're holding it
4. We do nothing, we do nothing and boom, our very intelligent Q learner says: Hey now is the time to sell and we continue like that through the rest of our time series

Let's now consider how these buys and sells affect our portfolio value (bottom graph)

- So we start out with whatever we've got in the bank and there's no trading for a while
- Suddenly there's a buy and then we see an increase in our portfolio value until we hit that sell (green line)
- Then a sell, red line
- Then nothing
- Then another buy (next green line) and a sell (next red line)
- :



- The real strategy however may not be this good, but ideally this is the kind of think we'd like to see

Quiz The trading problem: Rewards

It makes sense that rewards should relate in some way to the returns of our strategy. There are at least two approaches that we can utilize

- Short term rewards in terms of daily returns
- Long term rewards that reflect the cumulative return of a trade cycle from a buy to a sell
    - Or for shorting from a sell to a buy

Which results in faster convergence?

- **r = daily return (YES)**
    - *A reward at each step allows the learning agent get feedback on each individual action it takes (including doing nothing)*
    - If we reward a little bit each on each day, the learner is able to learn much more quickly because it gets much more frequent rewards
    - This is called immediate reward
- **r = 0 until exit, then cumulative return that we gained across that whole trade (NO)**
    - If we get no reward at all until the end of a trade cycle from a buy to a sell the learner has to infer from that final reward all the way back that each action in sequence there, must have been accomplished in the right order to get that reward
    - This is called delayed reward

Quiz The trading problem: State

Which of these factors make sense to be in a state?:

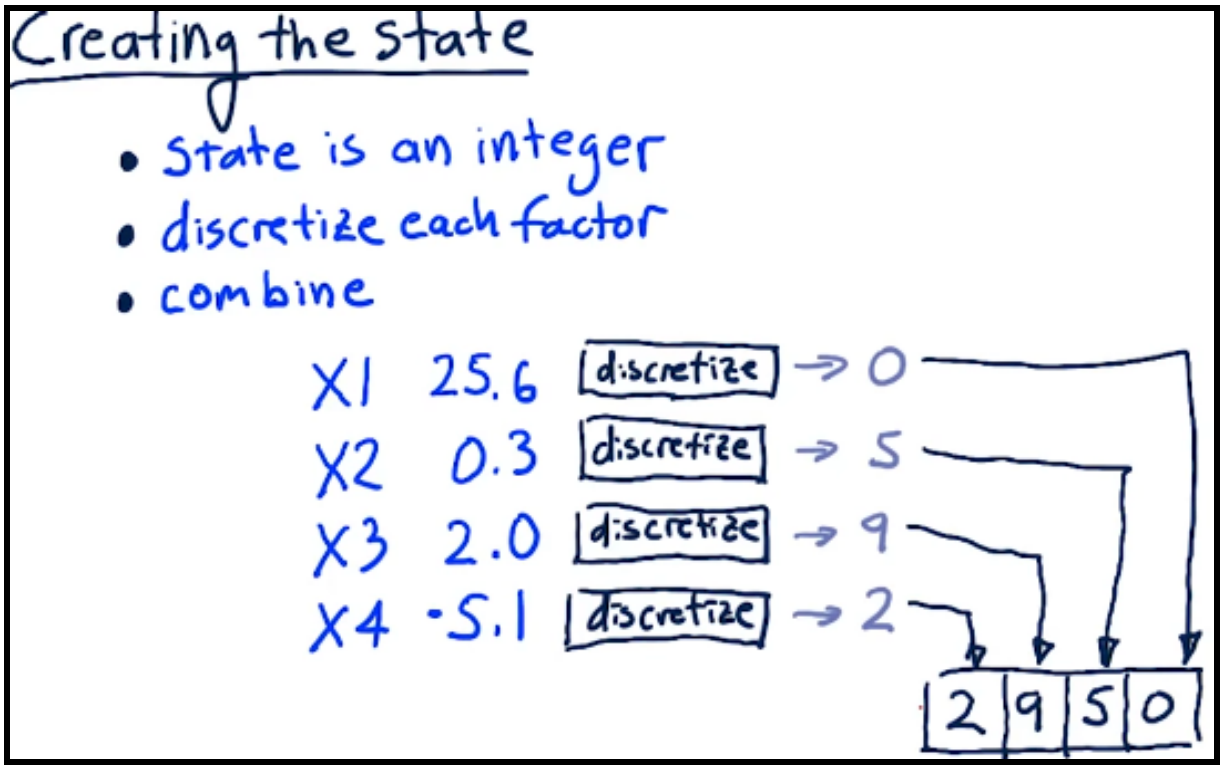
- **Adjusted Close (NO)**
    - Is not a good factor for learning because you're not able to generalize over different price regimes for when the stock was low to when it was high
    - Also if you're trying to learn a model for several stocks at once and they each hold very different prices adjusted close doesn't serve well to help you generalize
- **Simple Moving Average SMA (NO)**
    - Same thing applies as above in AC
- **Adjusted Close/SMA (YES)**
    - If you combine AC/SMA together into a ratio that makes a good factor to use in state
- **Bollinger Band Value (YES)**
- **P/E ratio (YES)**
- **Holding Stock (YES)**
    - This is an important for RL
        - If you're holding the stock it may be advantageous to get rid of it
        - If you're not holding it, you might not necessarily want to sell
    - So this additional feature about what your situation is, is useful
- **Return Since Entry (YES)**
    - This might help us set exit points for instance, maybe we've made 10% on the stock since we bought it and we should take our winnings while we can

Creating the State (this is encoding or hashing in a way)

- An important consideration in what we're doing here is that our state is a single number - it's an integer
- That way we can address it in a table, in our Q table

It is certainly the case that some RL methods are able to tread the state as a continuous value but just to get started here, let's use state as an integer so we can work more easily with the data.
Now we have to do a little bit of work to get our state to an integer and here is the general way to do it.

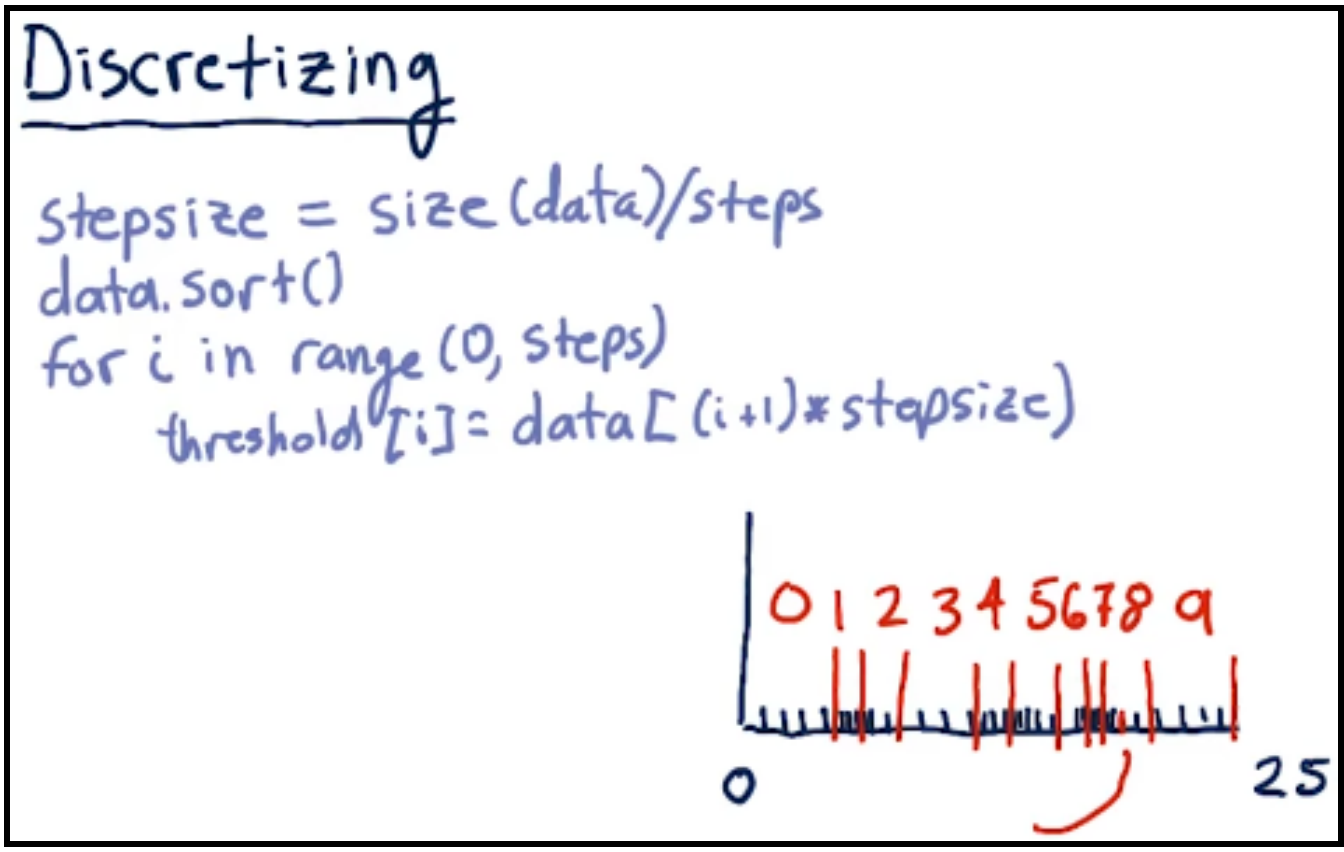
- The first step is to discretize each factor
    - It means convert that real number to an integer
- Next is combine all those integers together into a single number
    - We're assuming that we're using a discrete state space and that means more or less that our overall state is going to be in this one integer that represents at once all of our factors



- So consider that we have four factors and each one is a real number
- Now we have separately beforehand figured out how to discretize each of these factors
- So we run each of these factors through their individual discretizers and we get an integer
    - In the example he happened to select integers between 0 and 9 but you can have larger ranges, for instance.. 0 to 20 or 0 to 100 even
    - It's easy to go from 0 to 9 because we can then stack them together into a number but it's not too hard to think of algorithms that will enable you to combine larger ranges of integers together

Discretizing

Discretization or discretizing is an important step in this whole process. What we want to do is have a way to convert a real number into an integer across a limited scale.
In other words, we might have hundreds of individual values here between 0 and 25 of a real number and we want to convert that into an integer say, between 0 and 9

- First thing is we determine ahead of time how many steps (groups, bins) we're going to have
    - In other words how many groups (bins) do we want to be able to put the data into
    - To have an integer between 0 and 9, we would use 10 groups (bins) in this case
    - So we divide how many data elements we have altogether by the number of steps
- Then we sort the data and then the thresholds just end up being the locations for each one of these values
    - In other words if we had say 100 data elements and we were going to have 10 steps, our step size is 10
    - So we find the 10th data element and that's our first threshold, then our 20th, 30th and so on
- It ends up looking something like this:
    - When the data is sort of sparse, our thresholds are set far apart
    - When the data is not sparse, these thresholds end up being closer together
    - 

Q-Learning Recap

Remember, we're always training on before set of data and we then testing on later data

- **Building a Model**
  - Define states, actions, rewards
    - States are combinations of our features
    - Actions are buy, sell, do nothing
    - Rewards are some type of return
  - Choose in-sample training period
  - Iterate Q-table update
    - And iterate over that training period and update your Q-table on each iteration
  - Backtest
    - When you reach the end of that training period, you backtest to see how good the model is and you go back and repeat until the model quits getting better
    - Once it's converged you stop and you've got your model
- **Testing a Model**
  - Backtest on later data

Summary

Advantages

- The main advantage of a model-free approach like Q-Learning over model-based techniques is that it can easily be applied to domains where all states and/or transitions are not fully defined.
- As a result, we do not need additional data structures to store transitions `T(s, a, s')`or rewards `R(s, a)`.
- Also, the Q-value for any state-action pair takes into account future rewards. Thus, it encodes both the best possible *value* of a state (`max_a Q(s, a)`) as well as the best *policy* in terms of the action that should be taken (`argmax_a Q(s, a)`).

Issues

- The biggest challenge is that the reward (e.g. for buying a stock) often comes in the future - representing that properly requires look-ahead and careful weighting.
- Another problem is that taking random actions (such as trades) just to learn a good strategy is not really feasible (you'll end up losing a lot of money!).
- In the next lesson, we will discuss an algorithm that tries to address this second problem by simulating the effect of actions based on historical data.

Resources

- CS7641 Machine Learning, taught by Charles Isbell and Michael Littman
  - Watch for free on Udacity (mini-course 3, lessons RL 1 - 4)
  - Watch for free on YouTube
  - Or take the course as part of the OMSCS program!
- RL course by David Silver (videos, slides)
- A Painless Q-Learning Tutorial

03-07 Dyna

- One problem with Q learning is that it takes many experience tuples to converge
- This is expensive in terms of interacting with the world because you have to take a real step, in other words execute a trade, in order to gather information
- To address this problem, Rich Sutton invented Dyna

➢ Dyna works by building models of **T**, the transition matrix, and **R** the reward matrix
➢ Then after each real interaction with the world, we hallucinate many additional interactions, usually a few hundred that are used then to update the Q table

Dyna-Q Big Picture

➢ Dyna-Q is an algorithm developed by Rich Sutton, intended to speed up learning or model convergence for Q learning
➢ Remember that Q-learning is model free, meaning that it does not rely on **T** or **R**
    ○ T being the transition matrix and R being the rewards function
➢ Dyna ends up becoming a blend of model free and model based methods

Here is how it works

Let's first consider plain old Q-learning:

➔ Init Q table
➔ Observe s
➔ Execute a, observe s', r
➔ Update Q with experience tuple <s,a,s',r>
➔ Repeat to observe s

Dyna is really an addition to Q learning

When we augment Q-learning with Dyna-Q, we had three new components

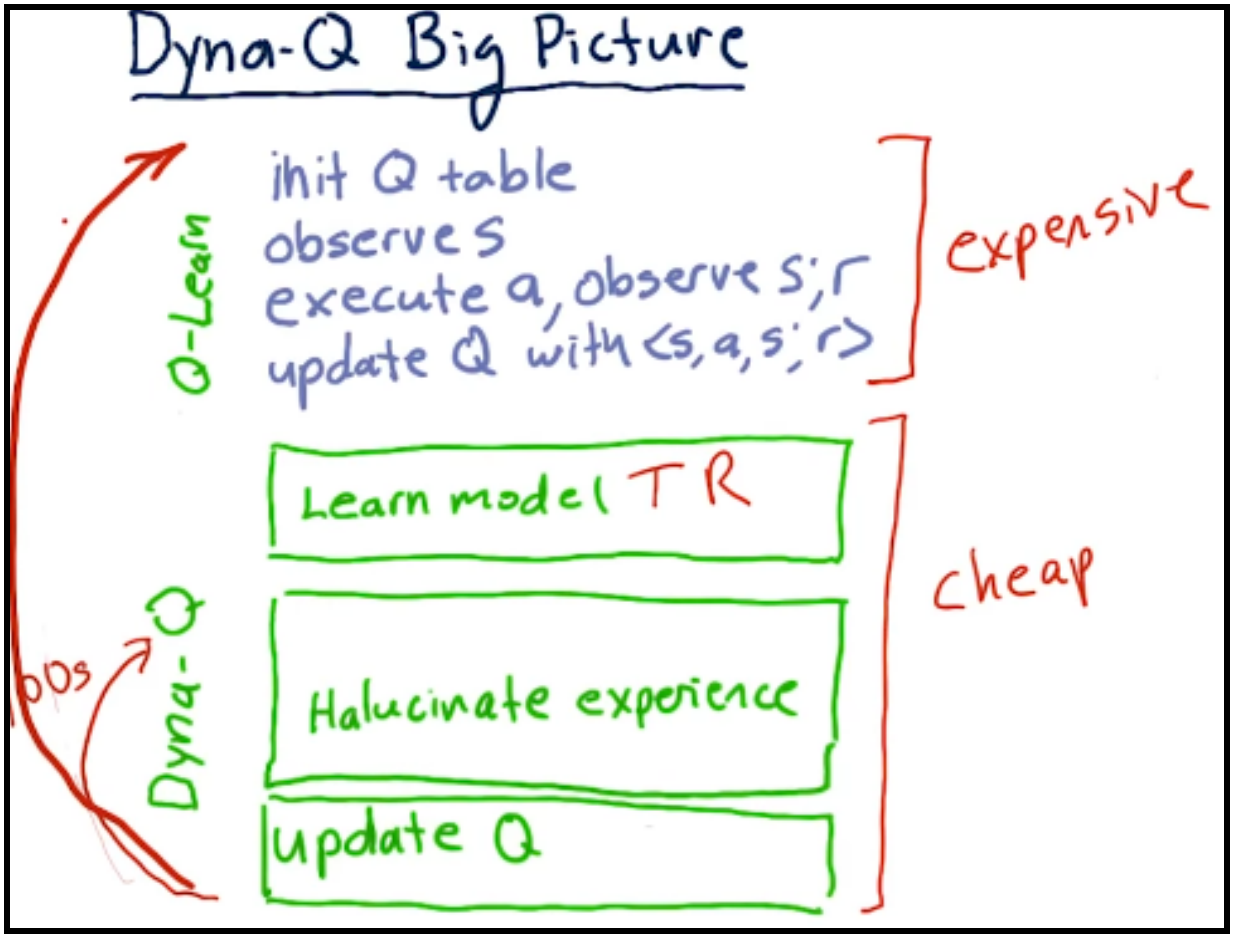
1. Learn Model
    a. We add some logic that enables us to learn models of T and R
2. Hallucinate experience
    a. Rather than interacting with the real world like we do appear with the Q-learning part and (which is expensive by the way) we hallucinate these experiences
3. Update Q
4. Repeat this many times, hundreds of times

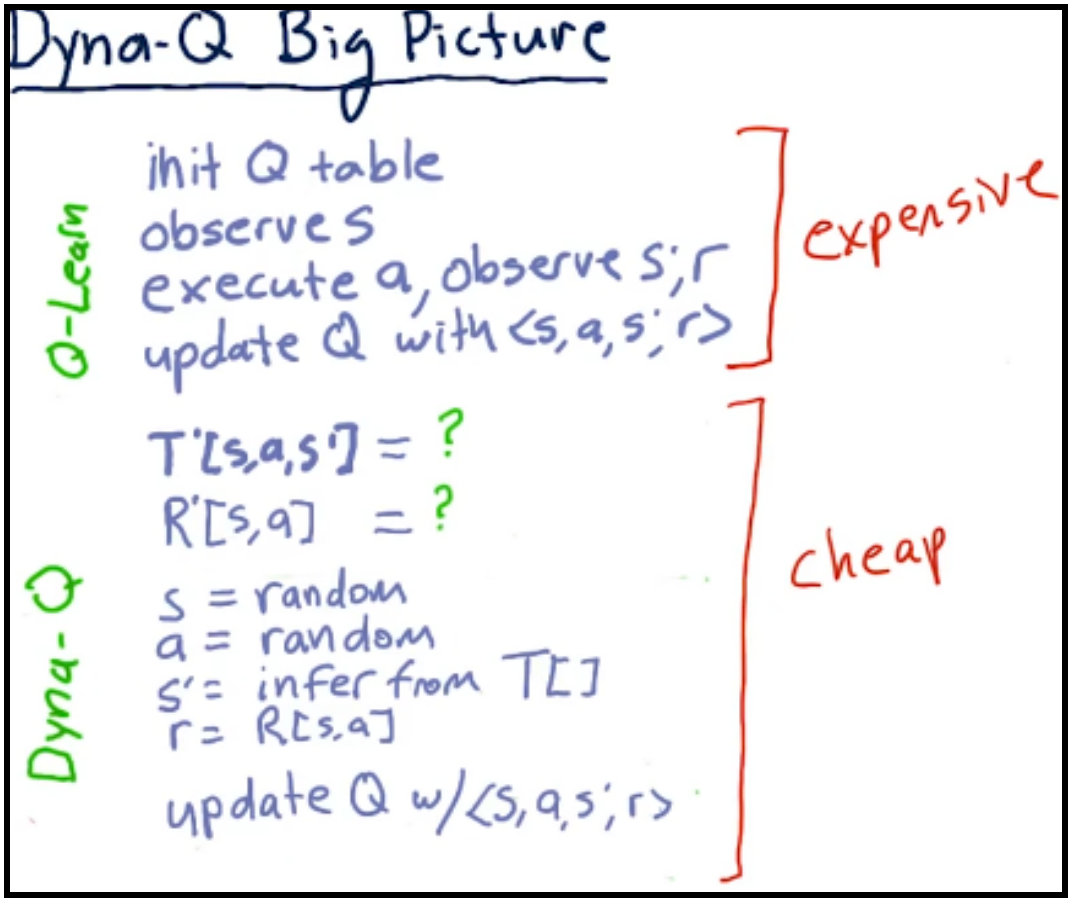
The Dyna-Q operations is very cheap compared to interacting with the real world because we can leverage the experience we gain in the Q-learning part from the interaction with the real world, but then update our model more completely before we step out and interact with the real world again.

After we iterated enough times, 100, 200 times, then we return back to resuming our interaction with the real world at the **Observe s** step.

The key thing here is that for each experience with the real world we have maybe 100 or 200 updates of our model here.

1. Learn Model T, R
    a. What we really want to do here is find new values for T'[s,a,s'] and R'[s,a]
    b. The point where we update our model includes the following
        i. We want to update T, so it's called T' for the moment, which represents our transition matrix and we want to update our reward function
        ii. Remember that T is the probability that if we are in state s and take action a we end up in s' and R is our expected reward if we are in state s and take action a
2. Hallucinate Experience
    a. First we randomly select an s
    b. Second we randomly select an a
    c. Then we infer our new state s' by looking at T
    d. And we infer our immediate reward r by looking at big R or the R table
3. Update Q
    a. Since we have the <s,a,s',r> experience tuple from step 2 and we can update our table using that

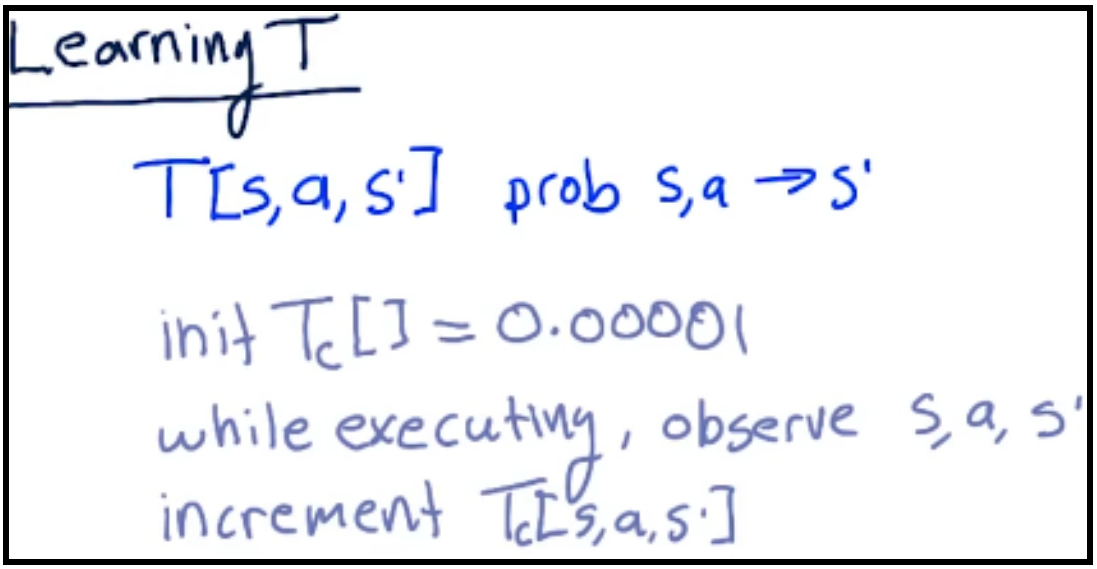

Dyna-Q Big Picture
Q-Learn
init Q table
observe s
execute a, observe s', r
update Q with <s,a,s',r>
expensive
Learn model T R
Halucinate experience
update Q
cheap
100s
Dyna-Q


Dyna-Q Big Picture
Q-Learn
init Q table
observe s
execute a, observe s', r
update Q with <s,a,s',r>
expensive
T'[s,a,s'] = ?
R'[s,a] = ?
s = random
a = random
s' = infer from T[]
r = R[s,a]
update Q w/<s,a,s',r>
cheap
Dyna-Q

Learning T

These may not correspond exactly to what Richard Sutton did, but these methods work well in practice.

- Remember that T[s,a,s'] represents the probability that if we are in state s, take action a, we will end up in state s'
- To learn a model of T we're going to observe how these transitions occur
    - So in other words we'll have experience with the real world, we'll get back a s,a,s' and we'll just count how many time did it happen
- We're introducing a new table called Tcount or Tc
    - We initialize all of our Tc values to be a very, very small number, if we don't do this we get in a situation where we have a divide by zero
    - Then we begin executing Q learning and each time we interact with the real world we observe s, a and s' and then we just increment that location in our Tcount matrix
    - So every time we see it transition from s to s' with action a, boom we add one and that's pretty simple



Quiz How to evaluate T in terms of Tc?

Remember Tc is just the number of times each of these has occurred

You simply need to normalize the observed count Tc[s,a,s'] of landing in next state s' by the total count of all transitions from state s on action a, i.e. summed over all possible next states.

Type in your expression using MathQuill - a WYSIWYG math renderer that understands LaTeX.

E.g.:

- to enter Tc, type: T_c
- to enter Σ, type: \Sigma

For entering a fraction, simply type / and MathQuill will automatically format it. Try it out!

Correction: The expression should be:

$$\frac{T_c[s,a,s']}{\Sigma_i T_c[s,a,i]}$$

In the denominator shown in the video, ***T*** is missing the subscript ***c***.

Tc[s,a,i] is Tc[s,a,:]

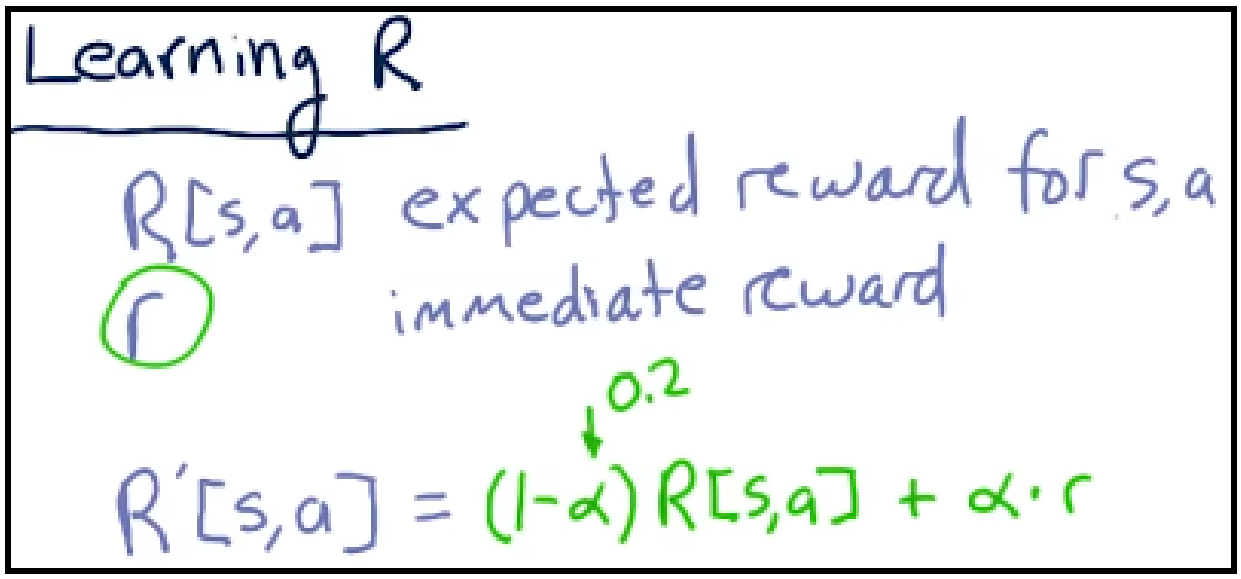
Learning R

Remember when we execute an action a in state s, we get an immediate reward, r

- R[s,a] is our expected reward if we're in state s and we execute action a
  - **Is our model**
- r is our immediate reward when we experience this in the real world
  - **Is what we get in an experience tuple**

We want to update this model every time we have a real experience and it's a simple equation, very much like the Q table update equation:

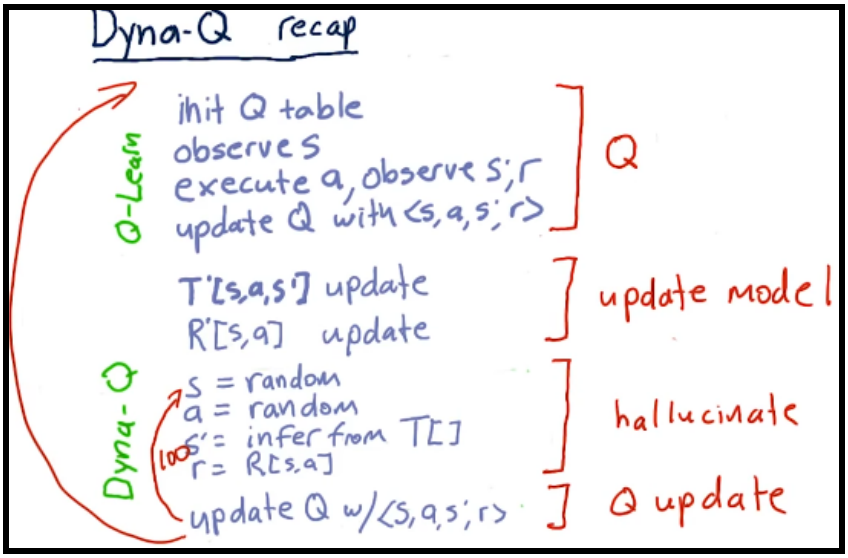
- What we have is one minus alpha where alpha is our learning rate, typically 0.2, times our current value for R[s,a] and then we add in our new estimate of what that value ought to be, alpha times r
  - We just use r for that estimate



- So we're weighting presumably our old value more than our new value so we converge more slowly.

Dyna-Q recap

We start with straight regular Q-learning and then we add three new components:



The three components are

- We update models of T and R
- Then we hallucinate an experience
- And update our Q table

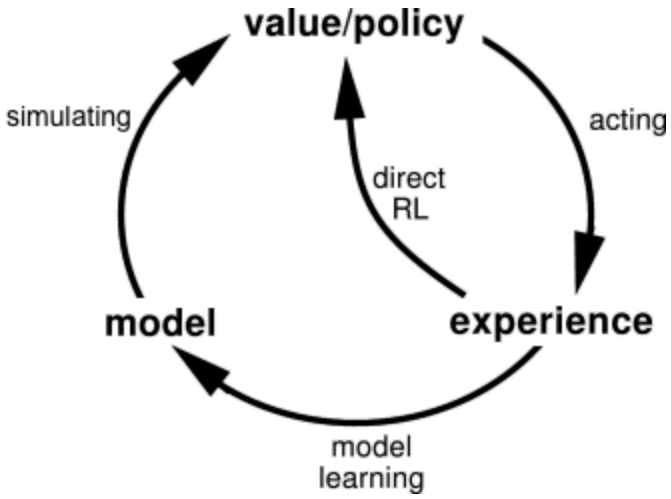
Now we repeat this many times, in fact maybe hundreds of times until we're happy. Usually 100, 200 times. Once we completed these steps, we then return back up at the top and continue our interaction with the real world, the top is observe s not init Q table.

The reason Dyna-Q is useful is that these experiences with the real world are potentially very expensive and these hallucinations can be very cheap. And when we iterate doing many of them, we update our Q table much more quickly.

Summary

The Dyna architecture consists of a combination of:

- direct reinforcement learning from real experience tuples gathered by acting in an environment,
- updating an internal model of the environment, and,
- using the model to simulate experiences.



Sutton and Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998. [web]

Resources

- Richard S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, Austin, TX, 1990. [pdf]
- Sutton and Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998. [web]
- RL course by David Silver (videos, slides)
    - Lecture 8: Integrating Learning and Planning [pdf]

Interview with Tammer Kamel

Tammer Kamel is the founder and CEO of Quandl - a data platform that makes financial and economic data available through easy-to-use APIs.

Listen to this two-part interview with him.

- Part 1: The Quandle Data Platform (08:18)
- Part 2: Trading Strategies and Nuances (10:53)

Note: The interview is audio-only; closed captioning is available (**CC** button in the player).

Options

Exchange Traded Options

- What they are
- What it means
- How to read an options chain
- What are call and put options
- What are some basic strategies that you can employ with options
- What's the purpose
- How do you price options
- How can you combine different options contracts in order to get more advanced strategies that let you take a specific position in the market based on a very specific opinion that you have about what's going to happen that can go well beyond just long or short
- *There are no details of the black-scholes model or how to trade volatility smile or how to manage your positions over time rolling out and down if your position goes against you*

Options in this chapter are exchange traded stock options and not employee stock options. CBOE or Chicago Board of Exchange are such examples.

An option is a legal contract which gives you as the buyer the right but not the obligation to buy (I think this should be both Buy and Sell) the underlying stock at a specific price on or before a specific expiration date.

- For the first part of the talk we're assuming we're talking about buying ***call options*** only, which means ***buying the right to buy the stock***
- There are options on many other underlying assets besides stocks, there are options on:
    - Future contracts
    - Commodities
    - Currencies
- When we say we're going to be able to ***buy the stock at a specific price***, we call that the ***strike price***
- When we say on or before the expiration date we're talking specifically US style stock options
    - In European style options we can execute the option on the expiration date only. Under EU style, you can't choose to exercise your option prior to the expiration date, but in US style you can

Option Chain

Let's look at an option chain to try to make all of this real, so we have a common position to go forward with. The below is the options chain for call options on Apple (AAPL) stock with a December expiration date:

Option Chain for Apple Inc. (AAPL)

Calls	Root	Strike	Last	Net	Bid	Size	Ask	Size	Vol	Open Int
Dec 16, 2016	AAPL	101				0		0	0	19
Dec 16, 2016	AAPL	102	9.05			0		0	0	2
Dec 16, 2016	AAPL	103	7.95			0		0	0	10
Dec 16, 2016	AAPL	104	7.90			0		0	0	25
Dec 16, 2016	AAPL	105	7.16	0.11		0		0	26	10856
Dec 16, 2016	AAPL	106	5.90	0.30		0		0	140	65
Dec 16, 2016	AAPL	107	5.26			0		0	0	26
Dec 16, 2016	AAPL	108	4.43	0.68		0		0	119	540
Dec 16, 2016	AAPL	109	3.49			0		0	0	215
Dec 16, 2016	AAPL	110	2.73	-0.17		0		0	769	64871
Dec 16, 2016	AAPL	111	2.10	-0.14		0		0	115	2660
Dec 16, 2016	AAPL	112	1.55	-0.15		0		0	407	6762
Dec 16, 2016	AAPL	113	1.20	-0.04		0		0	159	2058
Dec 16, 2016	AAPL	114	0.76	-0.02		0		0	65	1824
Dec 16, 2016	AAPL	115	0.53	-0.06		0		0	587	228847
Dec 16, 2016	AAPL	116	0.41	0.01		0		0	15	1124
Dec 16, 2016	AAPL	117	0.23	-0.01		0		0	68	2596
Dec 16, 2016	AAPL	118	0.20	0.02		0		0	3	1681
Dec 16, 2016	AAPL	119	0.12	-0.05		0		0	2	296
Dec 16, 2016	AAPL	120	0.09	-0.01		0		0	48	52997
Dec 16, 2016	AAPL	121	0.12			0		0	0	254
Dec 16, 2016	AAPL	122	0.09			0		0	0	115

We can look through the options chain example and see what some of the different data points mean and how we can interpret it.

- The first column, **Calls** is the **Expiration Date**. So all of these options expire Dec 16, 2016 - if you were to buy one of these options, that would be the date by which you would have to use or exercise the option or else it just simply goes away
- The second column, **Root**, this is just jargon that means the **underlying security**, in our case the **stock** that this option is tied to
- The third column, **Strike**, is the strike price we mentioned earlier, this is the price that you will be guaranteed to be able to buy the stock at if you choose to exercise the option.
    - **The key to these options is you can choose to buy at this strike price no matter what the real price of the root security is on the day you exercise the option that's kind of where the power and flexibility comes from**
- The fourth column, **Last** is because we took this after market close on that day, so it's showing the last price that each of these options traded at on that day as the market closed.
- The fifth column, **Net**, is simply showing the change from the previous time period

- The sixth and seventh, **Bid Ask** and **Ask Size**, are all 0 because the market was closed so of course there were no open orders at that moment
    - If you look at this during the day you would see some order book information about the current **Best Bid** and **Best Ask** price were for each one of these
- The eighth column, **Open Int (open interest)**, is kind of interesting, this shows how many total positions of this option are open and in the market right now
    - One thing you'll notice about this that is kind of interesting is humans like nice round numbers
    - The interest is much higher for no really special reason necessarily in all of the prices that end in a multiple of 5, just a little bit of humans being humans

So what are we doing with these option contracts?

An option contract controls 100 shares of the underlying stock, it's always 100, never more never less - but when we look at these prices, option contracts are always priced per share - this is either a historical artifact or salesmanship to make options look cheaper and it's how it's always been done.

- So let's say we look at the $110 strike price for AAPL, and we see that it costs $2.73 ,
    - what that means is if I buy a single 110 call option on AAPL,
    - that option will control 100 shares of AAPL stock
    - and my actual price that I pay for the option will be $2.73 * 100 so this option will actually cost $273 if we wish to buy it
- So we can buy one contract of the AAPL Dec 16 2016 call option at 110 dollars strike price for $273 which is $2.73 a share
    - Then anytime the market (Chicago Board of Exchange) is open before Dec 16 2016 I can choose to exercise this option
        - If we exercise this option, whoever sold me the option is obligated to sell me 100 shares of AAPL stock at exactly $110/share no matter what the price of AAPL stock is on that day.
        - So by itself buying this call option amounts to a bet on our part that AAPL stock will rise to at least 110 by the expiration date
            - (almost, because we actually paid money for the option and when we're calculating our break-even price, we really should factor in the money that we paid to buy this option because that is a lost expense and we don't get that money back)
                - So we really need for AAPL to rise to $112.73/share so that we earn back the $110 that we would spend buying the stock if we exercise the option and the $2.73/share that we spent to get the option in the first place.
                - Then if the price of AAPL on that day is $112.73, we break even and if it's anything above that we make money by exercising the option
                - If it's anything below that, we would lose money by exercising the option
                - We are ignoring trading fees in this *lecture* because that complicates things in unnecessary ways

So what is the point in buying the option, why wouldn't I just buy the stock directly?

There are advantages and disadvantages.

Advantages

- We will look at a specific scenario, using AAPL and $110 strike price with $2.73 value
- On the day that this presentation was first prepared, APPL stock was trading at $111.57

❖ So AAPL today: 111.57 and we're considering what to do, **let's say we buy stock**:
  - ➢ We think that AAPL will go up in the near future, we could buy 100 shares of AAPL today, it's a nice round lot, your broker will like you, that's how you minimize your commissions and that will cost us $11,157. That money disappears from our account and we get 100 of AAPL stock in exchange.
  - ➢ Say the price of AAPL does go up like we hope and it rises to 120 a share some time on or before the theoretical expiration date of the options that we're talking about but didn't buy this time, we just bought the stock, then we could sell our stock for $12,000.
    - Even if we don't sell, that is the paper amount that we're worth and we can still calculate a paper gain and then we have a profit on this particular trade of $843 - trading expenses which is pretty good for just holding the stock for 2, 3 weeks

❖ **Now let's say we buy an option contract** and the option contract we looked at was the $110 call and that would cost ust 273 to buy now.
  - ➢ Now in the **first situation** if AAPL rises to 120 by the expiration date, then we can choose to exercise our stock option which allows us to buy 100 shares of AAPL at 110 strike price we agreed on when we bought the contract, so buying that stock on that day costs us 11,000 and then we can immediately sell the stock the same day at the open market price for 12,000 so doing that our profit is 727 which is a little bit less, but we got to participate in most of the upside move in the stock and you can already see one of the benefits that we obtained by using the option instead of buying the stock and that is we only tied up $273 in our account and the rest of the cash was still there in the account available to use
    - compared with if we had had bought the stock, we would actually have to tie up the full 11,157 and it would not be available to do anything else with during that time
    - So this is the leverage aspect of options and one reason that we like them because you can control more money using less money
  - ➢ Now in the **second situation** the AAPL stock doesn't go to the way we expect
      - AAPL today: 111.57
      - BUY 100 shares: -11,157
      - AAPL falls to 100 by Dec 16 2016, SELL: +$10,000
      - P/L: -$1,157 which is a pretty big loss for just a few weeks
    - If we bought the option, we buy that 110 call for $273 and then we assume again that AAPL falls to 100 and it stays there until the option expires, we simply don't exercise the stock option.
      - Remember this is an option, that's the entire point - we're buying the right but not the obligation to buy the stock - we don't have have to use it if we don't want to

and if we're below our breakeven price, we won't, we'll just let the option expire and at least it won't do any further damage to us. So in that case our loss is just the 273 that we spent buying the call option, no matter how low AAPL stock price falls

- ➢ If we want to make the numbers even more extreme, imagine that something terrible happens and AAPL goes out of business, then if you bought the stock you lose every single dollar that you had in it - if you bought an option, you lose exactly the premium that you pay for the option and it doesn't matter what happens to the stock after that

These are some of the reasons why we like options:

- **First we cannot lose more than the premium that we paid for that option up front, no matter how far it moves against you, you can't possibly come out worse than losing the premium you pay for it**
- **We like it because leverage, we still get most of the upside potential if the stock goes the right way, not quite all but most and in exchange we tapped our downside to the premium and we've tied up a lot less money in our account so we could go out and take more bets and diversify our opinion about the market in more directions.**

Now it's obviously not all good, if it were just this then you would say why should I ever buy the stock? Instead of just always trading in options - and there are some people who do that, but **there are downsides to options**:

- **First the premium is lost money, it's gone and you pay it immediately to another person when you acquire the option contract and you don't get it back no matter what happens to the stock**
- **Second, options have expiration dates, this is a biggy. You're adding a layer to the bet that you're making in the stock market**
    - If we just go buy some stock, I'm betting that that stock will go up eventually. I'd like it to go up sooner, rather than later but it's ok if it doesn't. I can just hang on to the stock and forget about it and someday in the future when the market goes back up I can sell the stock at a profit.
        - Yes I've tied up money for longer
        - Yes I couldn't close out the position maybe when I wanted to because I was in the red, but I never have to sell it, I can just hold on and hope for the best
- **With an option you can't hold on, options have a built-in time bomb**
    - You're now taking a bet that the stock will move in a specific direction by a specific amount or more and in a specific usually pretty short time period
        - So there's a lot of benefits but you're also now making a much more specific bet that's more likely to be wrong because you've got timing and all these other elements that you don't have if you just buy the stock
- **With options you don't own the stock**
    - And this sounds kinda trivial, but there are implications to the fact that when you buy an option to buy the stock, you've not yet bought the stock, so the person who sold you that stock option, still owns the stock until you choose to exercise it.
        - That means, while the stock price is going up before you exercise the option, they have the voting rights that might come from that stock, not you, and they are collecting the dividends and any other benefits of ownership of the stock, not you

The **moneyness** of an option

- One of those words you don't hear in casual conversation and it's not proper english, but it's common financial term

Why are some options so much more expensive than others?

If you look back here at the option chain that we're studying, for example down at the bottom of the page, we've got the 117, 118 options that are only 20, 23 cents per share but then at the top the the 102 strike price option is currently trading at 9.05. So if we were to buy that for for 100 shares, that would cost $905 for one contract, whereas the 117 would only cost you $23 for 100 shares.

➔ **Why is that?**

- ◆ The answer is, right now as of the day that we did this, AAPL is trading at 111.57 so AAPL was trading right there in the middle of the table between those 111 and 112 strike prices
  - If we were to buy that 110 call option that we keep looking at for 2.73/share we could exercise that option right now, one moment after we buy it for a profit of about 1.57/share because AAPL right now is already trading 1.57 above the 110 strike price
    - So the option is not obviously going to be priced less than that 1.57/share that we could get if we bought the option, exercised it and sold the stock immediately, all in one go
      - ◆ **That is what we call the intrinsic value of the stock:**
        - **The difference between the option strike price and the underlying spot price for an *in-the-money option* is called the intrinsic value of the stock**
        - **In-the-money is a little recursive, but it's an option that currently has a non-negative intrinsic value so you could make money immediately by buying the option, exercising it and then selling the stock**
      - ◆ **Something that doesn't have intrinsic value is called an out-of-the-money option - meaning currently you could not exercise that option at a profit because of where the stock price is right now and then if you're right on the border between those two, you can be at the money or near the money or any other language that you want to wrap around that**

★ So we say **intrinsic value** right now is 1.57 /share for 110 strike price because that's what we could get if we sold it right now, immediately after buying it, but it's not priced at a 1.57/share, it's 2.73 so what is the rest of that price?

- The 110 costs 2.73 and has intrinsic value of 1.57,
- The 115 has an intrinsic value of zero because AAPL is trading less than that right now, so it's not currently worth anything in the sense of exercising the option, but it still costs 53c/share or $53 for the contract of 100 shares, so there must be some other component to this.

        - This other component is what we call the **time value** of the option. The more time there is between now and the option's expiration, the better the chance there is that the stock will reach the strike price before it expires, and so the higher the time value will be
        - **The excess premium cost of an option beyond its intrinsic value is attributed to time-to-expiration**
            - We can think of this as the amount of time that the stock has got to move the right way before the option expires, and we'll see of course that the further we are from the expiration date, the bigger that time value will be and as we get closer to the expiration date, the time value shrinks, and it shrinks especially rapidly in the last few days and really the last few hours before the expiration as it becomes nearly impossible that the stock will reach each strike price

With options, you're not stuck with them once you buy them - these are not something that you buy and then there's no market for it to sell them second hand, these can be bought and sold just like stocks. So if you change your mind on the option after you bought it you can sell it to someone else via an exchange like the Chicago Board of Exchange - but you'll probably lose money if the stock hasn't gone your way fast enough because of time decay.
If you hold this option and the price hasn't moved or it's not moving in the correct direction, the time value of the option will be decreasing, because there's less time before the expiration date for it to make that move that you need on and out of the money option. **This is also called Theta.** There are many different greek letters used to describe different secondary characteristics and derivatives and second derivatives of various factors of options related to to stocks.

- If you want to trade options you have to go study them, but this is the only one that we'll mention.

Theta or time decay, is the rate at which an option is losing it's time value, which is to say it's the first derivative of that time value over time.

- The time values changes more rapidly as the expiration date arrives and the time decay goes up
- Then when you get to the expiration date, of course time value approaches zero and all of the option prices will approach their intrinsic value because you will have to buy it, exercise it and sell it immediately at that point when it's just about to expire
- **The last two weeks is when time decay goes the fastest.** So people who want to actively trade options, as opposed to hoping to exercise them and get the stock, usually try to close out their open option positions at least two weeks before the expiration date
    - So the smart money is really not in there at the last second trying to do option trades that are about to expire.
    - They're buying those options a month, or two months out and then selling them as you get to four, three, two weeks away from expiration onto somebody else who actually wants to make that bet about whether the stock will end up in the right place and they'll get to exercise the option.
- If all of this part about the Greeks and time decay is really interesting to you, you may want to go look at the **Black-Scholes model**, which is not machine learning related - but it is one of the very common financial models to use to decide what should the price of an option be at any given moment, and a lot of people use deviations from that model to try to do arbitrage on options where they moved away from the expected price and you might expect some reversion to the mean

We made an assumption so far, about the options that we're talking about - and we mentioned briefly in the beginning, but we'll state it again because then we're going to relax that assumption and talk about some other types of options.

Everything we've talked about so far has assumed that we are buying a call option.

- So a call option is equivalent to a long position
    - It's going to profit if the underlying stock goes up and you have been buying it - which means that you're buying the right to buy the stock

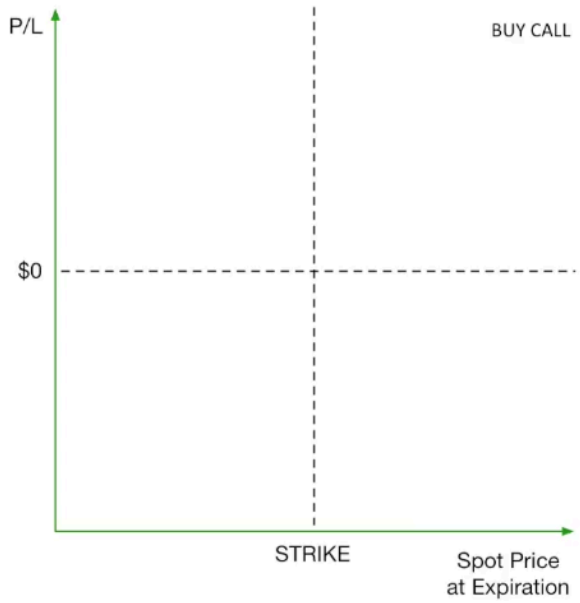
There are several other basic things you can do with options even before we get to the complex strategies. We will go through those while showing a **profit and loss curve** in options and that makes it very easy once you get accustomed to looking at the curve . Doing this will help us understand exactly what an option strategy is doing, and particularly to understand exactly when it will make money and when it will lose money.

These are really helpful visualizations:

Call options

The following is the basic grid that we'll always use for this profit and loss curve for an option strategy.

- ➢ At the bottom we will have the spot price at expiration of the underlying security
- ➢ In the case of our example, this bottom axis would represent what is AAPL's stock price at the time the option expires and we'll draw a line vertically in the middle of it at the strike price.
    - It doesn't actually matter what the numbers are here, when we visualize these things, we're just saying under this strategy, **where is the spot price at expiration, relative to the strike price of your option and that will give you the appropriate shape of your profit and loss curve, even without filling in the precise strike prices and dollar amounts.**
- ➢ On our vertical axis, we'll have profit and loss again, with zero kind of in the middle where we break even on the entire thing and we will include our premium as part of that.
    - **If we're above that, we're making money, and if we're below that we're losing money**

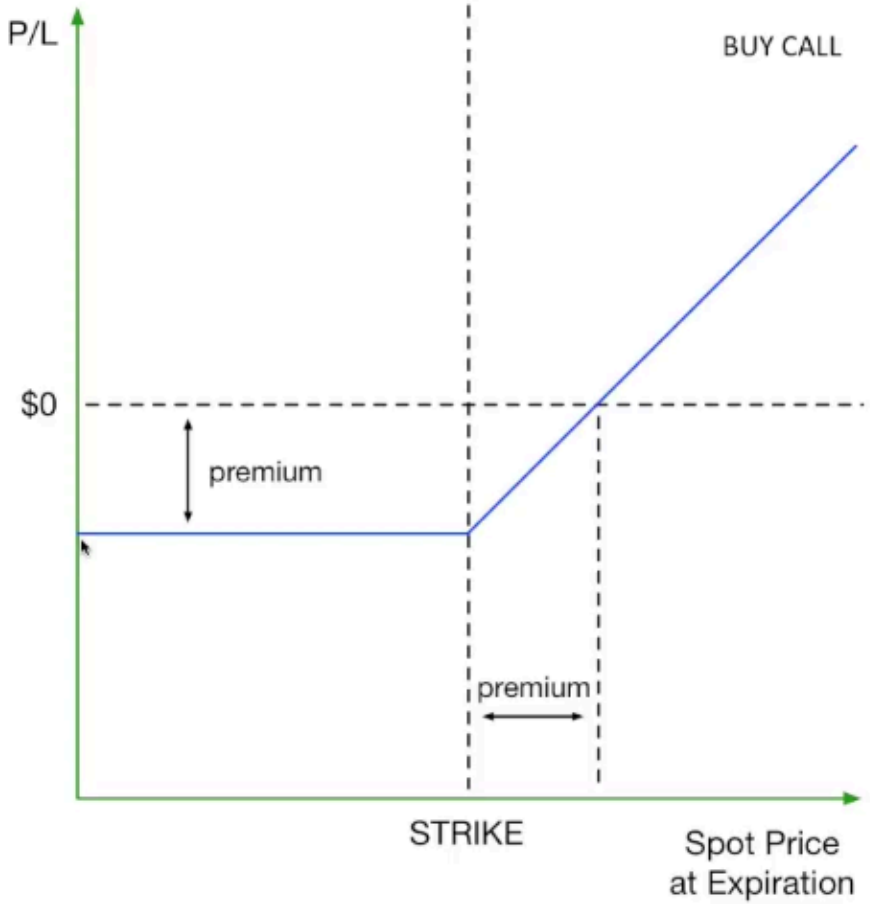


P/L is profit and loss

To understand this further, let's look at a simple buy call which is what we've been talking about up to this point.

- So when we buy the call, we've already said that we're paying some premium upfront and that's money that I'm in the hole immediately as soon as I buy the option. And then I need for the stock to rise to some amount above the strike price that represents that extra premium that I paid before I actually broke even and recovered all of the money that I spent, plus been able to sell the stock and get back to zero.

If we look at a P/L curve for a standard buy call:



- When we buy the call, we've already said that "I'm paying some premium upfront" and that's the money that I'm in the hole immediately as soon as I buy the option
- Then I need for the stock to rise to some amount above the strike price that represents that extra premium that we paid before we've actually broken even and recovered all the money that i spent, plus be able to sell the stock and get back to zero.
- If we look at the P/L curve for for a standard buy call:
    - We start in the left end, with the spot price well below the strike price, so in other words the stock price went down, doesn't matter how much, it just went down quite a bit below the strike price and no matter how far down it goes, i've lost a dollar amount that exactly correlates to the premium that I paid and that's going to be the case until I get to the strike price
    - As soon as we hit the strike price, we're still losing money, but we'll start losing less because once we hit that strike price, we will exercise the option because we'll make at least a little bit of money exercising it, buying the stock at the cheaper price that I have, and then selling it at the market price
        - In fact this is one dollar for one dollar: for every dollar that we go above the strike price, we'll make one dollar per share of profit not counting that premium.
            - This is just a simple linear relationship
        - And once we have exceeded the strike price by the premium, we finally reach out actual P/L break even point where now we will be made whole on this entire trade, we got back to 0, and we haven't lost any money.
        - And as we continue to rise above that, we're actually finally in a truly profitable position having bought this call option.

But buying calls is far from the only thing we can do, and kind of further away we get from it, the more interesting it will be, so we will talk about some of the other things that we can do with options.

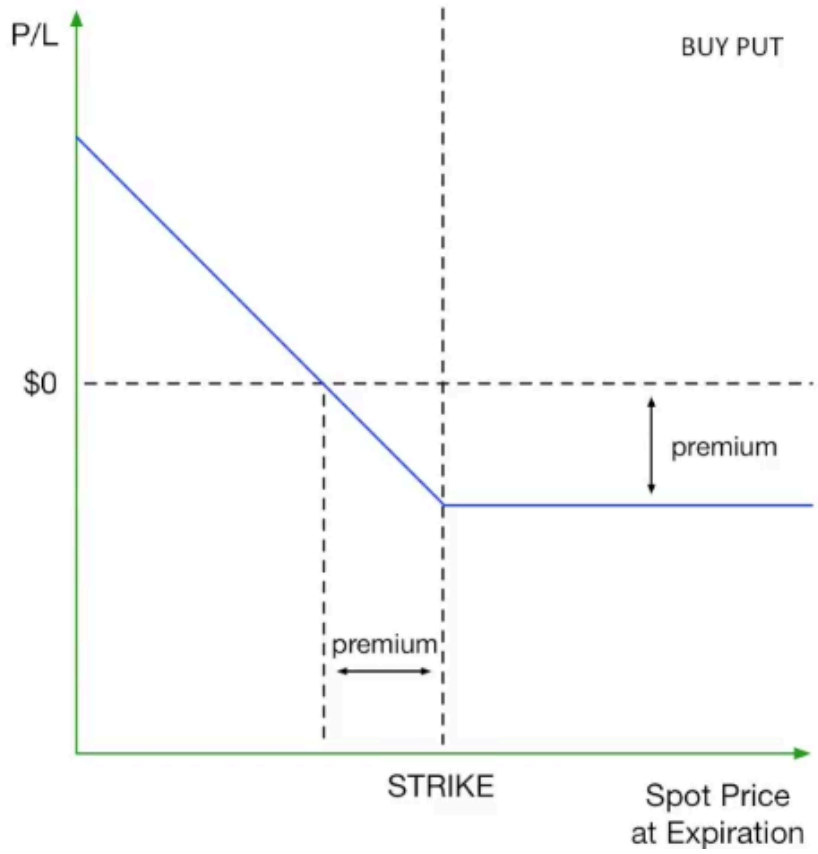
- As we go through each strategy, we will highlight the maximum profit and maximum loss we can experience with each strategy because that is one useful thing to think about
- Here on the buy call, is very simple, but do notice that the maximum loss I could possibly experience is my premium amount which we had said before
- The maximum profit from buying a call is theoretically unlimited because there's no amount where the spot price of the underlying stock would have to stop going up
    - It could go up any amount, and I profit dollar for dollar along with it, with this position once I exceed my breakeven point

Put Options

The other basic type of exchange-traded stock option that you need to know about is the put option. The put option is essentially the opposite of a call option. You'll buy it just the same, and you'll pay a premium when you buy it just the same, but now instead of buying the option to buy the stock, you're buying the option to sell the stock.

- You don't have to own the stock, to be able to do this, because you're probably going to have a margin account if you are playing with options anyway, and you can short the stock as part of your exercising of it and then immediately buy it back to close out the position

So if we look at a **buy put**, we would expect the P/L curve to be opposite in some sense, and indeed it is:



- Because now we're making a bet that the stock will go down and we'll be able to sell it, or short sell it, at the strike price and then immediately buy it back to close the option position and the stock position at whatever the price is on that day, so we'll be able to sell it for more, and then immediately buy it back for less and pocket the difference.
- So our cost curve is going to be almost exactly the opposite
    - If the stock closes above the strike price, we will experience our maximum loss which will be the premium that we paid for this put option
    - If we get down to the strike price, then we'll start recovering some of our money again, one dollar for one dollar, every dollar that the underlying spot price goes below the strike price, will earn back a dollar of our premium that we spend until we finally get up to a true break even point where we've recovered the full premium
        - And then for every dollar that the stock falls below that, will earn a dollar profit per share that's covered by our option contract

There is one important difference to note though, and that is with a put option, we actually have an axis over here to run into, so our maximum loss is still limited to our premium, but our maximum profit is no longer unlimited, because the lowest that the stock could go is zero dollars a share.

- ❖ So there is going to be some limit which will essentially be the **strike price - premium** that will be our maximum profit, because again we can't hope for the stock to do worse than go to zero, stop trading and the company go out of business and that will limit the maximum profit we could make

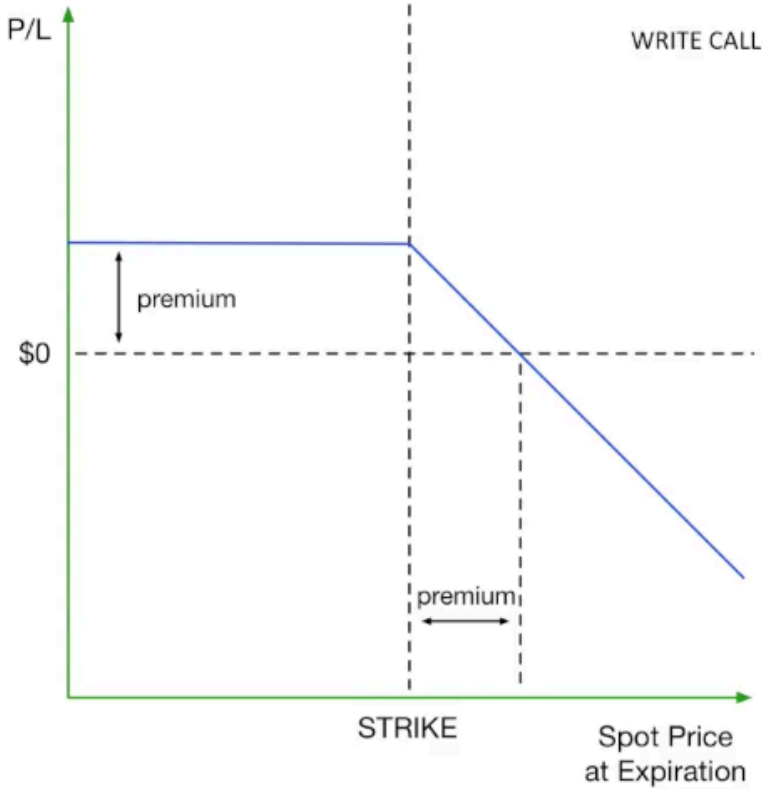
There are a couple of other things that you can do even with basic options that we will talk about next. Everything so far has assumed that we're buying an option, but if we're buying options - someone must be selling options and there is no reason why we could not be the person selling the option

instead of buying it. Typically in options, this is called writing the options contract - so let's look at what happens to the cost curves if we write options instead of buying options.

To explain writing a call:

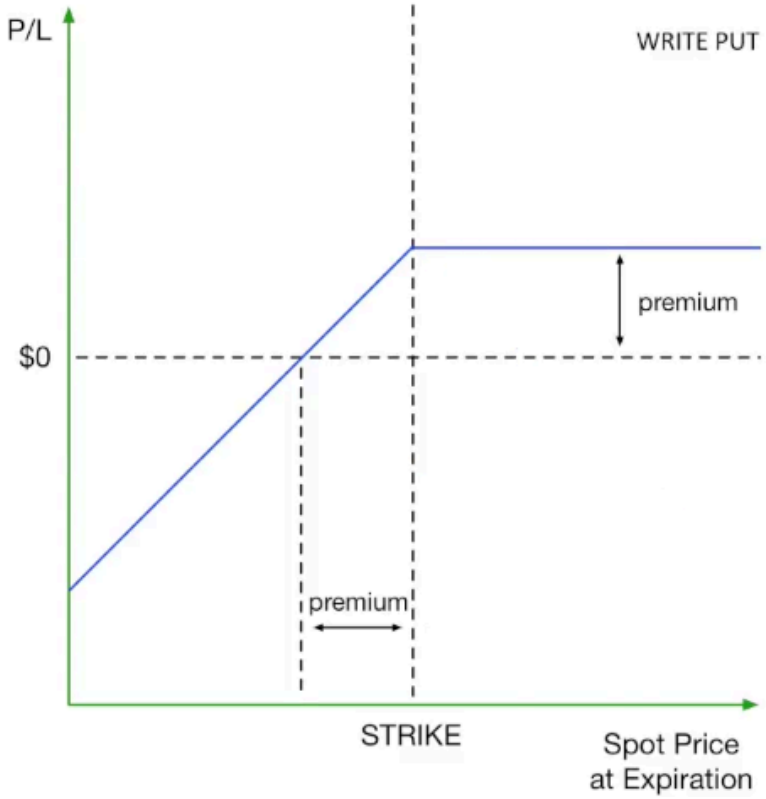
- it is the exact opposite in the other way between a call and put. When we write an option contract we are selling someone the option, then until the expiration date they can force me to buy or sell them the stock at the strike price whether or not I want to and no matter what the price of the underlying stock is on that day.
- For example with a call, I am selling someone the call option, I collect that premium payment immediately and I get to keep it no matter what, but then until the expiration date, they can buy our stock away from us, at the strike price no matter what.
- Now the best case they won't exercise that option, we got their premium to keep and we keep our stock, but of course the downside is that we might be forced to make a very unprofitable trade at some point prior to the expiration date if things don't work out well.

So here we might expect that the P/L (profit and loss curve will end up upside down of where the buy call curve was, and we'd be right, if we write a call we flip the buy call cost curve over:



- So the first thing that happens is that we now collect the premium for writing this call option, and that's cash in our pockets, so as long as the stock expires below the strike price, meaning it would be unprofitable to exercise the option for the person who bought it from us, we have a profit equal to that premium we collected and they will let the option expire.
- Once we get to the strike price, then the person who bought the option is earning dollar for dollar because now they will exercise the option since it's above the strike price and they're gaining a little bit back dollar for dollar and we're now losing dollar for dollar, because they're going to exercise that call and take our stock away from us at a disadvantageous price.
    - But we're still sort of okay until the stock price gets to the strike price plus the premium and at that point we are actually overall losing money on having written this option because now the person on the other end is going to exercise it, take it away from us at a very disadvantageous price compared to the current price of the stock and our premium we collected was not enough to make up the difference.

- **Writing calls (which are called naked calls)** are if this is our sole position and we're just doing it without any other strategy attached to it - and that's meant to suggest that it's probably a bad idea, and here's why:
    - Our maximum profit is now limited to the premium, there's no way we can make more money than the premium
        - This can be ok if the odds of that happening are very very high - but here's the slightly scary part of it, our maximum loss is now unlimited
        - Strike axis doesn't really count for putting a floor because it is just the profit and loss axis, we're not saying the stock price goes to zero, that is our loss
        - So now the stock price could now go up without any limit and the more it goes up the more money we lose, so we have to manage a strategy like this very carefully to ensure that we don't lose essentially some arbitrary unlimited amount of money
- Writing a put action actually does not have have that same unlimited loss potential, because we'll flip the cost curve back the other way so it'll be left to right opposite of writing a call and it will be top to bottom opposite of buying a put.



So if we write a put, we now have a floor under us, which is that the stock can't go down any lower than the zero, so when we write a put, we give someone else the option to sell us the stock at the strike price should they choose to do so.

- So now we want the price of the stock to go up, because we don't want the person to choose to sell us the stock because if they do that, it will mean they're winning and we're losing
    - So as long as it stays above the strike price, then we've got the premium that we charged in our pocket as profit and that is our maximum profit
    - Once the stock falls below the strike price, we start losing money dollar for dollar
        - We break even when it's the strike price minus the premium and then we start actually losing money on the trade at least with the floor at some point when the stock hits zero and we can no longer lose any money beyond that point
        - **So our maximum loss here will be the strike minus the premium if the stock goes all the way to zero**

★ **The good news about writing options is that 90% of stock options that are bought, are not exercised.**

- While it may seem strange to take these trades where you've got potentially large losses and small fixed profits that are your premiums, we have to realize that the odds of these situations happening are not equal
- ***Most options expire unexercised, which means most of the time with writing an option you'll put that premium in your pocket and that's the end of the deal - the option will never be exercised - but you really have to watch out for that small percentage of the time when it is exercised because you can lose your shirt.***

So far we discussed the four basic things we can do with a single options contract:

➔ Buy a call
➔ Write a call

- ➔ Buy a put
- ➔ Write a put

- ★ But the real power of options is not in doing these naked calls or naked puts, or even just buying or selling puts, it's putting these things together with some other strategy to let you take a more complex position or to hedge your bets or to make a very precise bet on what you think it's going to happen in the market - next we're going to talk about a couple of these

Covered Call

- ★ One of the base strategies that is considered one of the most safe things you can do with options and in fact many brokers will allow you to do it even in a cash account when you haven't been approved for margin is the **covered call**.
- ★

The covered call is a very simple strategy that combines one option position with a position in the underlying stock, so **to enter the covered call position**, what you would do is

1. **buy a stock, and then**
2. **write a call** on that same stock

So we're giving someone else the privilege to be able to buy away the stock from us a stock that you do actually own.

We should look at the possible outcomes of this to understand the purpose of the strategy.

There are three basic possibilities:

1. Stock ends above strike before the expiration date:

- **"called away", sell at strike, lose profit above, profit at strike - purchase price + premium**
- This is not really ideal but it's not so bad.
- So if this happens, then our stock will be called away meaning that the person on the other end of that call option will exercise it and buy our stock from us below the current market price.
- So we'll be obligated to sell our stock at the strike price which means we miss out on any of the profit above that strike price if the stock continue to move because now we don't own it, the other person owns it
- So our total profit in that case is the strike price minus the original purchase price of the stock plus the premium that we collected when we wrote that call.
    - This is not the end of the world, we still made money, but but we didn't make as much money as we would have if we had just bought the stock and not written the call because then we would have gotten to fully participate in that profit as the stock moved up.
    - The other problem is of course we don't have the stock anymore - so if we really wanted to be in that stock because we think it's going to continue up, now we have to go buy again.
    -

2. Stock ends up, but below strike:

- **perfect, option not exercised**
- The stock could go up, but not up enough to hit the strike price of the option that we wrote and this actually the perfect thing that we want to happen because now the option will not be exercised because we're below the strike price and the person on the other end would lose money by exercising the option and we're in a perfect position now because the stock went up so we've gained money on our position in the stock and we still have the stock and can write another call or anything else that we want and we collected a premium that has accelerated our return on the stock that ended up not costing us anything

- so our profits so far on this stock that we still own is the current price of the stock minus the purchase price plus the premium that we collected for selling the option that was never used
    - This is where we would love to be with a covered call
    - We'd like to be in this middle range where we never actually have to sell the stock so we end up making the full same amount of money as if you just bought the stock and done nothing with options, holding the stock the whole time that it goes up, except periodically we're juicing those returns and getting more money that we would otherwise because we're writing calls and collecting premium that's just extra money to us and then hopefully it's not exercised and the stock keeps going up slowly, never hitting those strike prices and the premiums are just extra money that we put in our pockets the whole time

3. The stock goes down:

- **option not exercised, still have stock, loss-so-far:current price - purchase price + premium**
- If the stock goes down, the covered call still helps, it really only becomes a hindrance if the stock goes way way way up and we miss out on that stock move
- If it goes down the option won't be exercised and we'll still have the stock
- Of course we lost money because our stock went down, but our loss to date is the current price minus the purchase price of the stock plus the premium, just like the last case
- But realize what that premium is doing here, that has partially offset the loss that we would have had anyway form holding this stock during a downturn

So by writing these covered calls, we've gotten to hold the stock through a downturn and we can wait for it to go back up again and recover our money, but in the meantime during the downturn these premium payments we are collecting, are helping to offset the losses.

So a covered call can be a pretty good idea. We miss out on a little bit of upside potential if the stock takes off and goes through the roof and gets called away, but if it goes up less than that, then you collect money, make money on the stock and keep the stock and if it goes down, then we partially offset losses that we would have experienced anyway.

Married Put

We should know that there is a **similar strategy called the married put.** In which we buy the stock and then buy a put. So this is essentially hedging against downside moves.

- If the stock goes up, then we lose from our profit the amount of the premium we paid, basically for insurance
- If the stock goes down, then our loss during that downturn is going to be capped because once we hit the strike price, dollar-for-dollar your put will gain at the same rate that the stock loses value.
- So the difference between your purchase price and the strike price will put a cap on how much we can lose during this downturn

★ We might ask, why would we buy a put to turn our position into a married put, instead of just selling the stock if I think it's going to go down, and one of the main reasons here is that there can be a lot of tax implications from closing your stock position. Actually selling your stock for real, can trigger tax gains or tax losses or a wash sale or other positions we may not want to be in.

- By doing a married put, we can get most of the same benefits of selling our stock by being hedged through a downturn without actually having to close our stock positions and trigger a bunch of possible tax implications or other financial implications

A more complex options strategy

This was the final part of this basic overview of options. We will now look at a more advanced example of a complex option strategy that we could take on and the key that we should take from it is that options have more moving parts than stocks, the are more choices you have, there are more combinations you can make and so it gives us more flexibility to construct a more complex position than we could with just stocks.

The thought experiment here is we know what to do in the stock market if you think that a stock will go up, buy it, if you think a stock will go down, sell it if we're holding it or maybe short it if we feel very strongly about it. But what if we have some good reason to believe that the stock is going to trade sideways for quite a while? What if we think it's going to enter a calm period where nothing interesting is going to happen with the company and the stock is just going to flatline and go straight sideways, maybe with a little bit of noise but no too much - what do we do to profit from that? And if we're limited to just buying and selling stocks, the answer is basically you don't profit from that. A sideways market is kind of the death of stock grading because there's no volatility, there's not a lot of opportunities to find those peaks and throughs and buy and sell and make money, but with options we can construct a position that will specifically target maximum profit if the price of the underlying stock does not change at all - and that's kind of a cool example of the power of things we can do with options - so let's build one of these.

Butterfly

There are a lot of varieties of option strategies - now we'll have really fanciful names given to them by options traders. The first one is the **Butterfly. The Butterfly is a strategy that's specifically designed to profit when a stock or the market is going sideways**.

- So what are we going to do for a butterfly?
  - **AAPL is at 111, buy a 105 and a 115, write 2 110s (all CALLs)**
  - Let's use the prices that we had before, from the options chain, we said that AAPL was trading at 111, so what we would like to do is write two calls as close to the current price of the stock as we can and then we want to bracket that by buying a call below that and buying a call above that

Option Chain for Apple Inc. (AAPL)

Calls	Root	Strike	Last	Net	Bid	Size	Ask	Size	Vol	Open Int
Dec 16, 2016	AAPL	101				0		0	0	19
Dec 16, 2016	AAPL	102	9.05			0		0	0	2
Dec 16, 2016	AAPL	103	7.95			0		0	0	10
Dec 16, 2016	AAPL	104	7.90			0		0	0	25
Dec 16, 2016	AAPL	105	7.16	0.11		0		0	26	10856
Dec 16, 2016	AAPL	106	5.90	0.30		0		0	140	65
Dec 16, 2016	AAPL	107	5.26			0		0	0	26
Dec 16, 2016	AAPL	108	4.43	0.68		0		0	119	540
Dec 16, 2016	AAPL	109	3.49			0		0	0	215
Dec 16, 2016	AAPL	110	2.73	-0.17		0		0	769	64871
Dec 16, 2016	AAPL	111	2.10	-0.14		0		0	115	2660
Dec 16, 2016	AAPL	112	1.55	-0.15		0		0	407	6762
Dec 16, 2016	AAPL	113	1.20	-0.04		0		0	159	2058
Dec 16, 2016	AAPL	114	0.76	-0.02		0		0	65	1824
Dec 16, 2016	AAPL	115	0.53	-0.06		0		0	587	228847
Dec 16, 2016	AAPL	116	0.41	0.01		0		0	15	1124
Dec 16, 2016	AAPL	117	0.23	-0.01		0		0	68	2596
Dec 16, 2016	AAPL	118	0.20	0.02		0		0	3	1681
Dec 16, 2016	AAPL	119	0.12	-0.05		0		0	2	296
Dec 16, 2016	AAPL	120	0.09	-0.01		0		0	48	52997
Dec 16, 2016	AAPL	121	0.12			0		0	0	254
Dec 16, 2016	AAPL	122	0.09			0		0	0	115

- ➢ So in this case what we might do is we might buy a 105 call and and a 115 call as my bracketing pair and then in the middle, write two 110
- ➢ The cost of these is going to partially offset one another, making the butterfly cheaper than we might expect it to be for taking the position in four options, because we collect the premium for those 2 110 in the middle and we only pay the premium for the two on the outside that we're buying and in this case we're going to pay 7.16 for that 105 and pay 53 c a share for that 115, but we're going to collect 2*2.73 in the middle, so this whole position is going to cost us 2.23 a share to enter the butterfly or 223 because we're controlling 100 shares of stock underneath this
- ➢ **Premium: -7.16 + (2*2.73) - 0.53 = -2.23, thus cost to enter Butterfly : $223**

Let's look at the Profit and Loss curve for the Butterfly to understand what it's doing and why this is an interesting idea. A perfectly reasonable question to ask about this butterfly strategy we are going to look at is why is it called Butterfly and the answer is because it looks like a butterfly, lol, yolo.

The basic idea of the butterfly is, as we said,

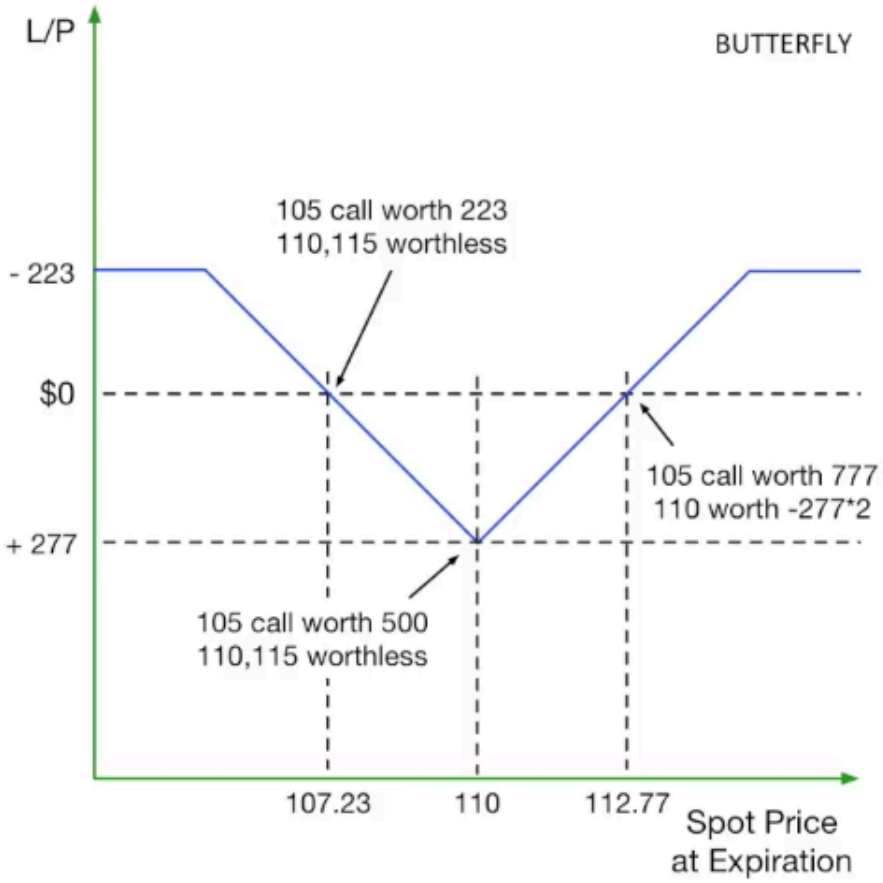
- We're going to gain money if the stock price doesn't move very far away from where it is now. That's the bet that we're making
- We're going to lose money if the stock price moves significantly in either direction so we're hoping for a period of low volatility

Please notice that the profit and loss axes on the left are inverted simply because it makes the butterfly look more like a butterfly, but don't let that fool us.

We should see why the different points on this graph work out the way we do. Intuitively it makes sense given the position that we took, that we would have our maximum profit exactly at that central strike price of the two

options that we wrote, that's our target price that we hope the stock will stay near and then our profit is going to taper off in both directions until we reach some maximum loss, because we've created a fully offset position with two writes and two buys. So once we get to an extreme of the stock price going up a lot, or down a lot, those two writes and those two buys are going to gain or lose value in lockstep which will put a cap on our loss and that's a good thing, it means we have a limited risk here.

So we'll notice our loss and our gain potential are about the same, that's not an accident, that's why we do this strategy.



- Starting from the left side of the graph, we are looking at a loss of -223 - everything we did was calls, so this makes sense if the stock price goes way way way down, then we're going to lose our 223 net premium that we paid to enter this total position, after we added up the premium of all of our options that we entered into.
- Once we reach the 105 dollar point, the first elbow, that 105 call that we bought, is going to start to be worth something. Below this everything was worthless. Our writes that we did won't get called away but we've already collected the premium and we've factored that into this total 223 it cost to get into the position.
    - So as soon as we hit 105, our 105 starts gaining a dollar for dollar and beginning to offset the loss of that premium
- Now our 110 is still worth nothing which is good because those were writes, and our 115 that we bought is also worth nothing. Once we get up to L/P == 0 and 107.23, then that 105 call is worth 223, because per share we're at 107.23 or 2.23 higher than the 105 call. Our 110 and 115 are still worthless, so at this point if we exercised the 105, we gained 223, that exactly offsets our 223 premium and we break even on this butterfly position.

- As the price continues to increase from there, the 105 continues to gain in value and so does our total profit until we reach L/P == +277 and 110 strike price.
    - This is where we would optimally love to be, at that point that 105 call is worth exactly 5 dollars a share or 500 total, and our 110 and 115 both still worthless, so we now have a net profit after paying our premium of 277 per option contract, for the stock price doing absolutely nothing - which is something that would be hard to achieve without an option strategy like that.
- After this we start losing money as the price goes up, so our 110 now unfortunately are worth something to the person we sold them to, and so they're going to be exercised and we're going to start losing money.
- Our 105 call will offset one of those 110 dollar for dollar and cancel it out but the other 110 is now going to start actually costing us money, dollar for dollar as the stock price continues to increase, until we get to the L/P==0 and 112.77 which is our second break even point.
    - At this point, 112.77, that 105 call is now worth 777, the 110 is going to work against us now and each are worth -277 per contract, (total 2*-277). And we notice that if we subtract those, we get 223 dollars. So the 105 is worth 223 more than what we're losing from the 110 - which means we've just barely offset our premium and we're breaking even again.
- Now if the stock price goes above that, then we're actually losing money because we haven't recovered our premium.
- Once we get up to 115, which is the last point before the graph flatlines, now our 115 call that we bought, has kicked in and it is gaining now value, dollar for dollar as the price of the stock continues to rise
    - So once we get there, the 105 and the 115 are both in the money and are fully offsetting those 2 110 that are losing - so every time the stock price changes a dollar, we earn a dollar on the 105, we earn a dollar on the 115, and we lose 2 dollars on the 110, meaning that we go straight sideways and our loss is capped

This was just one example of what we can do with a more complex option strategy, where we mixed together four calls, two buys and two puts, with cleverly offset prices, to construct an exact profit and loss curve that we're comfortable with when we think that the stock is going to go sideways.

We can make adjustments to this anyway we want. If we want a wider butterfly, we can give ourselves more room for the stock to move a little bit and spread out this butterfly by buying calls that are further away from those central writes, but we'll end up paying more in premiums to do it. **So the wider that butterfly gets, the more premium we're going to lose.**

We can also do something like unbalance the calls, so on the left hand side, instead of a 105 we could do 100, but still keep the other side at 115 - that let's us favor the stock moving in one direction over another, to just widen the butterfly in one direction. That's going to produce kind of a funny shape in the graph where one wing is higher than the other and traders actually call that a **broken wing butterfly,** because that's kinda what the P/L chart looks like and this is how everyone analyzes option strategies.

This was meant to provide a basic understanding of

- What exchange-traded options are
- What we should do with them
- How do they work mechanically
    - How we pay or collect premiums and the implications of that
- What moneyness is
- Intrinsic value vs the time value and the time decay
    - Which means we don't want to be holding these options when they enter those last two weeks before expiration, unless we really really know what we're doing
- A little bit of a look into more complex strategies

- The butterfly
- The broken wing butterfly
- The iron condors

All the things that people do to express an exact opinion about the stock market through the very flexible instrument that is exchange-traded options